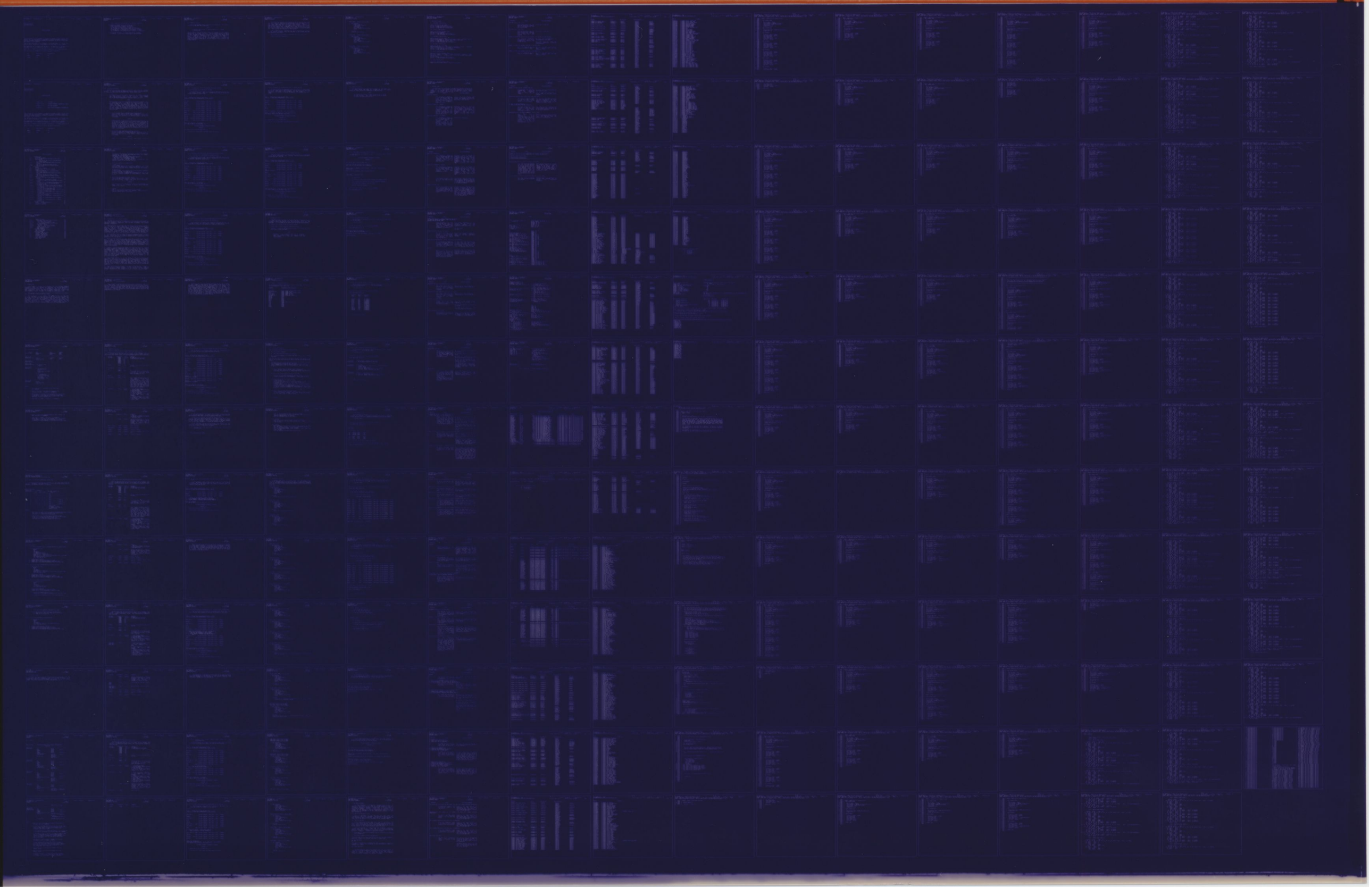
ENSAB-2.0

VAX 725/730
CUSTOMER RUNNABLE DIAG

EP-ENSAB-DL-2.0
1 OF 9 JAN 1986
COPYRIGHT© 1982-86

digital
MADE IN USA

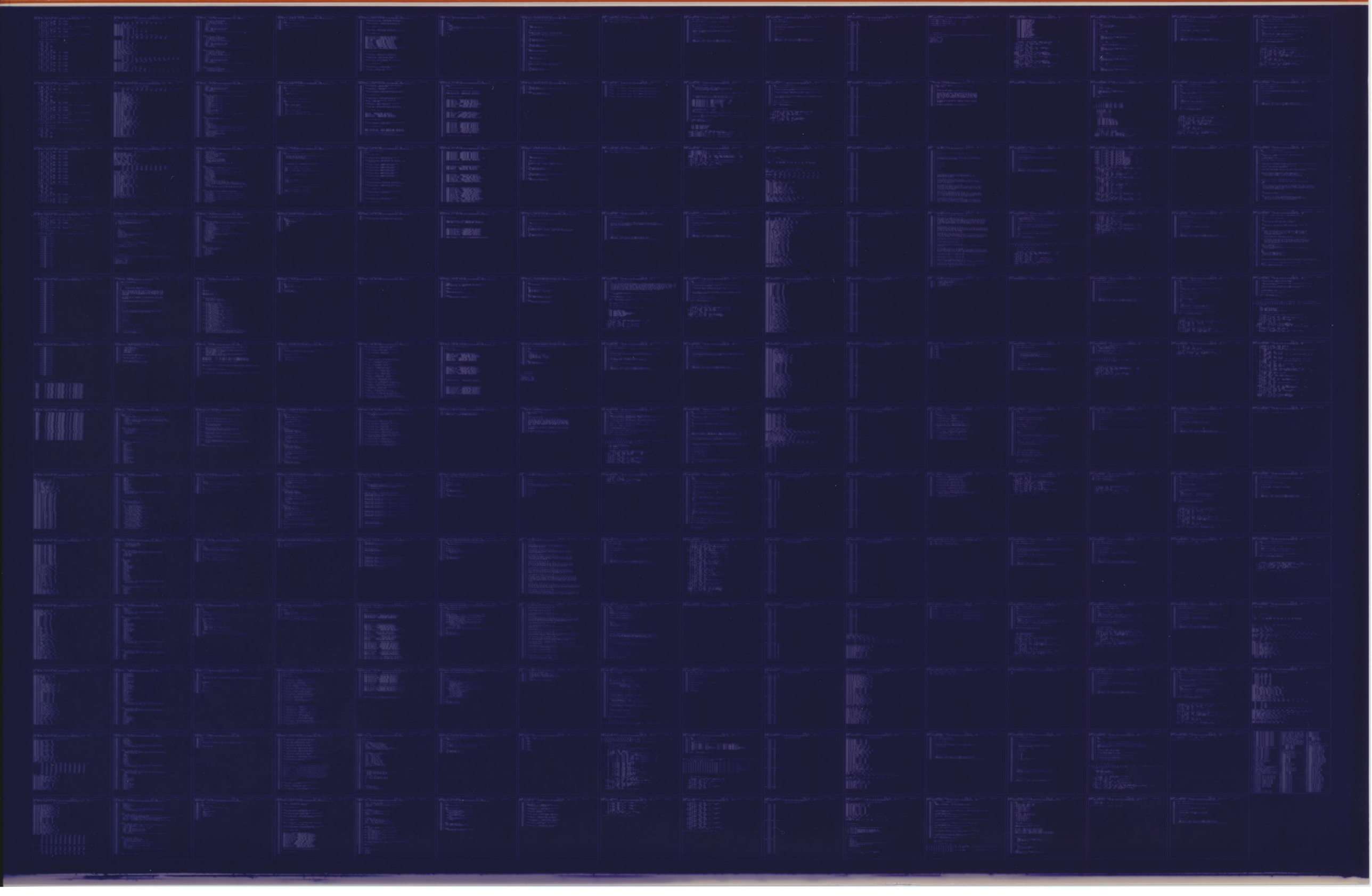


ENSAB-2.0

VAX 725/730
CUSTOMER RUNNABLE DIAG

EP-ENSAB-DL-2.0
2 OF 9 JAN 1986
COPYRIGHT© 1982-86

digital
MADE IN USA



ENSAB-2.0

VAX 725/730
CUSTOMER RUNNABLE DIAG

EP-ENSAB-DL-2.0
3 OF 9 JAN 1986
COPYRIGHT© 1982-86

digital
MADE IN USA

ENSAB-2.0

VAX 725/730
CUSTOMER RUNNABLE DIAG

EP-ENSAB-DL-2.0
4 OF 9 JAN 1986
COPYRIGHT© 1982-86

digital
MADE IN USA

ENSAB-2.0

VAX 725/730
CUSTOMER RUNNABLE DIAG

EP-ENSAB-DL-2.0
5 OF 9 JAN 1986
COPYRIGHT© 1982-86

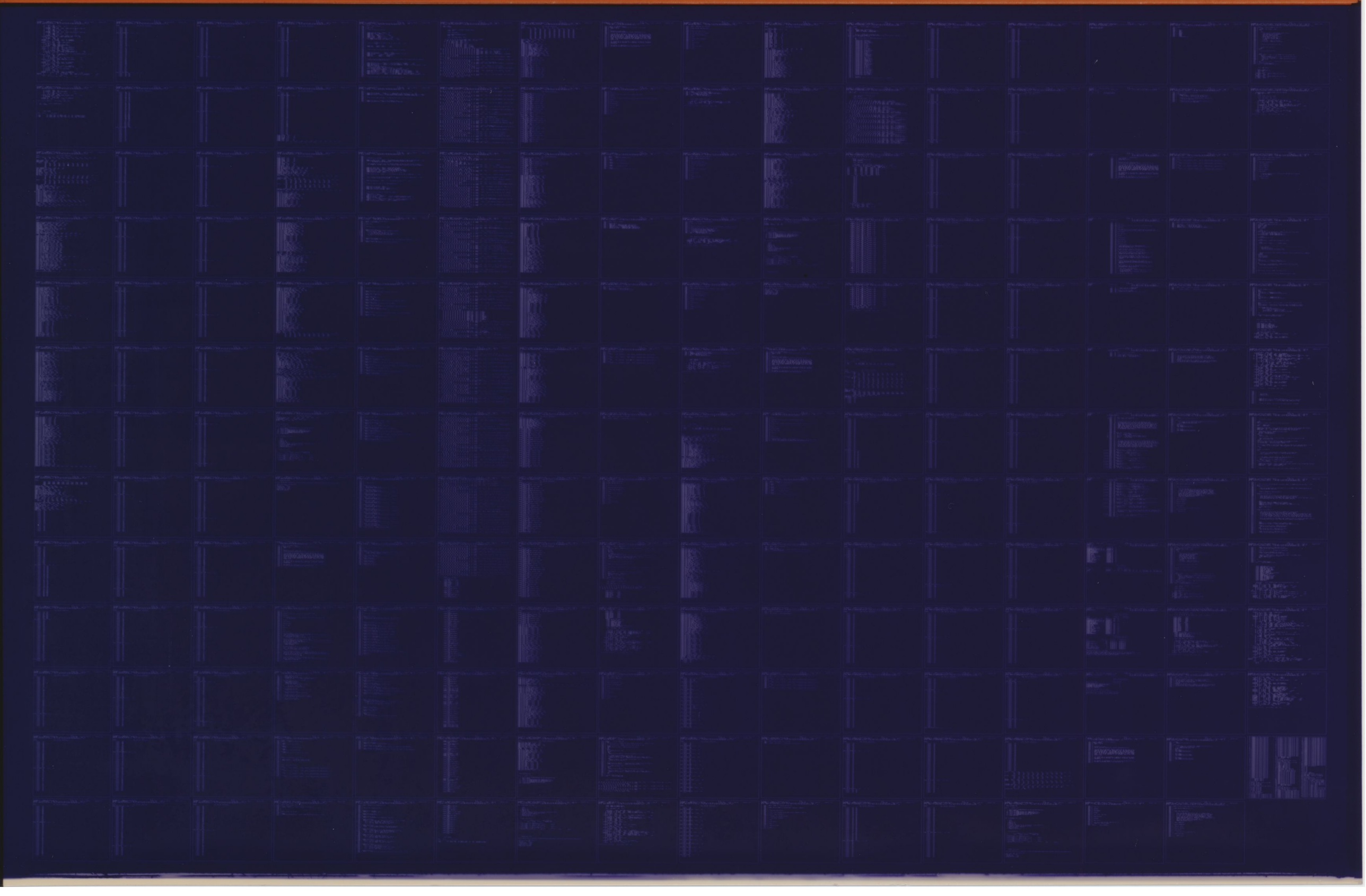
digital
MADE IN USA

ENSAB-2.0

VAX 725/730
CUSTOMER RUNNABLE DIAG

EP-ENSAB-DL-2.0
6 OF 9 JAN 1986
COPYRIGHT© 1982-86

digital
MADE IN USA



ENSAB-2.0

VAX 725/730
CUSTOMER RUNNABLE DIAG

EP-ENSAB-DL-2.0
7 OF 9 JAN 1986
COPYRIGHT© 1982-86

digital
MADE IN USA

ENSAB-2.0

VAX 725/730
CUSTOMER RUNNABLE DIAG

EP-ENSAB-DL-2.0
8 OF 9 JAN 1986
COPYRIGHT© 1982-86

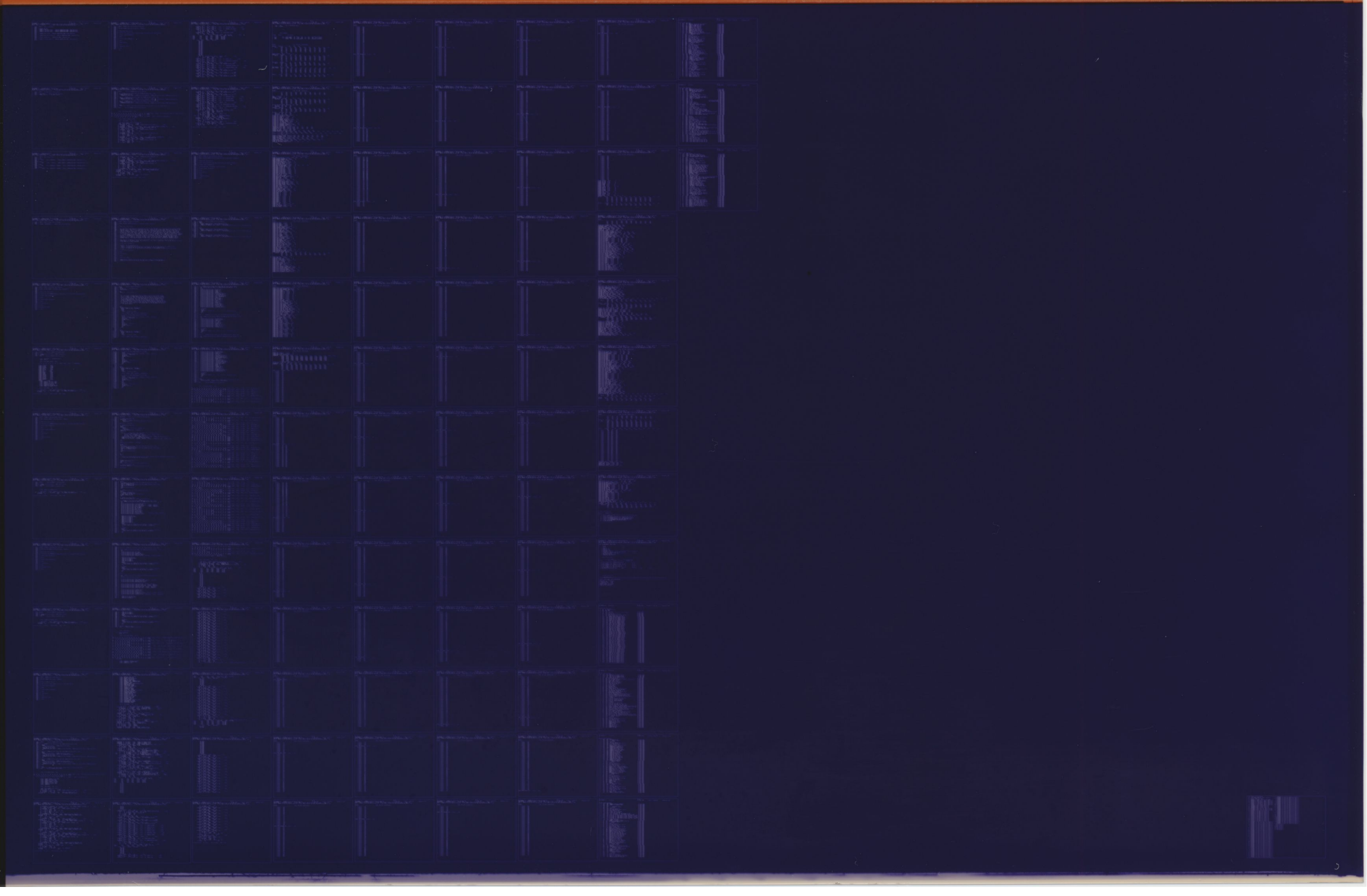
digital
MADE IN USA

ENSAB-2.0

VAX 725/730
CUSTOMER RUNNABLE DIAG

EP-ENSAB-DL-2.0
9 OF 9 JAN 1986
COPYRIGHT© 1982-86

digital
MADE IN USA



digital

IDENTIFICATION

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital or its affiliated companies.

Copyright (c) 1984 by Digital Equipment Corporation. All Rights Reserved.

The following are trademarks of Digital Equipment Corporation.

DEC	DECsystem-10	DECSYSTEM-20
DECUS	MASSBUS	PDP
UNIBUS	VAX	VMS

digital

IDENTIFICATION

Product ID: ZZ-ENSAB-1.5
Product Title: CUSTOMER RUNNABLE DIAGNOSTICS (CRD)
Document Title: CRD USER DOCUMENT
Maintainer: BASE SYSTEMS DIAGNOSTIC ENGINEERING

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital or its affiliated companies.

Copyright (c) 1984 by Dig'tal Equipment Corporation. All Rights Reserved.

The following are trademarks of Digital Equipment Corporation.

DEC	DECsystem-10	DECSYSTEM-20
DECUS	MASSBUS	PDP
UNIBUS	VAX	VMS

Table of Contents

1 INTRODUCTION 1
1.1 Distribution Media 2
1.2 Rules for Determining CRD Base System Type 4
1.3 Requirements 5
1.3.1 VAX-11/730 DUAL RL02 or RL02/R80 RB730 Disk-based Systems 5
1.3.2 VAX-11/730 TSU05 TS05/R80 RB730 Disk-based Systems 5
1.3.3 VAX-11/730 UDA/RA Disk-based System 5
1.3.4 VAX-11/725 LESI/RC25 Disk-based System 6
2 CRD AUTO OFF-LINE 7
2.1 Support 8
2.2 Initiation Procedure 11
2.3 Program Flow - 'T' Console Command 13
2.4 Diagnostic Testing 15
2.4.1 VAX-11/730 DUAL RL02 or RL02/R80 RB730 Disk-based systems 15
2.4.2 VAX-11/730 TSU05 TS05/R80 RB730 Disk-based systems 16
2.4.3 VAX-11/730 UDA/RA Disk-based systems 17
2.4.4 VAX-11/725 LESI/RC25 Disk-based systems . . . 18
2.5 Sample Printouts 19
2.5.1 Printouts for VAX-11/730 UDA/RA Disk-based Systems 20
2.5.1.1 Printout for a UDA kernel system. 20
2.5.2 Printouts for VAX-11/725 LESI/RC25 Disk-based Systems 21
2.5.2.1 Printout for a LESI/RC25 Disk-based system. 21
2.5.3 Printouts for VAX-11/730 RL02/R80 RB730 Disk-based Systems 22
2.5.3.1 Printout for system without the FPA option . 22
2.5.3.2 Printout for system where DMF32P and DMF32A are disabled 24
2.5.3.3 Printout for system where TU58 in drive 0 does not contain CRD AUTO Test 25
2.5.3.4 Printout for system where TU58 in drive 0 has missing or currupt files 26
2.5.3.5 Printout for system where user forgot to spin up RL02 pack 28
2.5.3.6 Printout for system where system pack has been write-protected 30
2.5.3.7 Printout for system where system pack is not spun up 31
2.5.3.8 Printout for system where error in RB730 testing 33
2.5.3.9 Printout for system where error in DMF32 testing 34
3 CRD MENU OFF-LINE 35
3.1 Support - Hardware/Diagnostic 36
3.2 Initiation Procedure 37
3.3 Diagnostic Testing 39

3.4 Sample Printouts 46
3.4.1 MAIN MENU (Functional Test) Printout 47
3.4.1.1 Exit Menu Test Printout 48
3.4.1.2 Print System Hardware Printout 49
3.4.1.3 Print Hardware Preparation Printout 50
3.4.2 TEST MENU Printout 51
3.4.2.1 TEST ALL - No Errors Printout 52
3.4.2.2 TEST ALL - With Errors Printout 53
3.4.3 CONTROL-C MENU Printout 54
3.4.4 FATAL ERROR Printouts 55
4 CRD ERROR DIAGNOSIS 57
4.1 CRD Error Message Printout 57
4.2 CRD Tracing Mode 57
5 REVISION HISTORY 58
5.1 September 1982 Release 58
5.2 January 1983 Release 58
5.3 September 1983 Release 58
5.4 November 1983 Release 58
5.5 March 1984 Release 58
5.6 June 1984 Release 58
6 MESSAGE INDEX 59

1 INTRODUCTION

This document is the main user document for the CUSTOMER RUNNABLE DIAGNOSTIC (CRD) Test Package and describes the USE and OPERATION of features associated with CRD. It is intended for the Field Service, or more experienced person familiar with elements associated with executing VAX Diagnostics such as the VAX Diagnostic Supervisor (VDS), the Microdiagnostic Monitor (MICMON), and related terms and expressions.

CRD supports two CPU systems in the Off-line (standalone) environment. They are the VAX-11/730 and the VAX-11/725.

CRD can operate in two modes: CRD AUTO Off-line and CRD MENU Off-line. CRD AUTO is an Auto test mode of CRD that verifies the operation of hardware for the VAX-11/730,725 base system including CPU, Memory, the VMS system disk, and the diagnostic load disk in the Off-line, or Standalone, environment. CRD MENU is a Menu-driven test mode of CRD which verifies operation of all subsystems including options for the VAX-11/730,725 in the Off-line environment. The Menu format provides a simple interface to select and test either one or all VAX subsystems.

1.1 Distribution Media

System Type	CRD Base MACRO-DIAG MEDIA	CRD MICRO-DIAG MEDIA	CONSOLE TAPE MEDIA
VAX-11/730 RB730/DUAL RL02	RL02#1 Disk Cartridge (note 1,2)	TU58#36 Tape	TU58#34 Tape
VAX-11/730 RB730/R80/RL02			
VAX-11/730 RB730/TS05/RL02	MT#1 Magnetic Tape (note 6)		
VAX-11/730 UDA/RA-Disk	A) For System with UDA/RA60 Drive 0 or 1: RA60#1 Disk Cartridge (note 1,2) B) For System with no UDA/RA60 Drive 1 or Drive 0, but Tape subsystem: MT#1 Magnetic Tape (note 2,3,4)		
VAX-11/725 LESI/RC25	RC25#1 Disk Cartridge (note 1,5)		

NOTES:

1) The first seven(7) characters of the Pack Label are 'CRDPACK'.

2) Media is the VAX-11/730 Complete Diagnostic Set released from SDC. Directory is [SYSMAINT].

3) Before CRD is initiated at the customer site, the contents of the Magnetic Tape must be transferred to directory [SYSMAINT] or [SYS0.SYSMAINT] on a bootable RA81, RA80, or RA60 disk located on UDA Drive 0.

4) Tape Label is 'CRDMT'. Directory is [SYSMAINT].

5) Media is VAX-11/725 complete Diagnostic Set released from SDC. Directory is [SYSMAINT].

6) Tape Label is 'CRDMT'. Directory is [SYSMAINT].
Before CRD is initiated at the customer site, the contents of the Magnetic Tape must be transferred to directory [SYSMAINT] or [SYS0.SYSMAINT] on the RL02 disk of the TSU05.

1.2 Rules for Determining CRD Base System Type

The following Controllers can co-exist on the same VAX-11/730,725 system. Therefore, when CRD executes, it will use the following rules to determine the CRD Base System it is testing.

VAX-11/730,725 Controller Configuration	CONTROLLER IDC	CONTROLLER LESI	CONTROLLER UDA	CRD BASE SYSTEM TYPE (note 1)
1	X			IDC
2		X		LESI
3			X	UDA
4	X	X		IDC (note 2)
5	X		X	IDC (note 2)
6	X	X	X	IDC (note 2)
7		X	X	LESI or UDA whichever is found at UNIBUS I/O Fixed CSR address 772150

NOTES:

1) IDC refers to either of the two VAX-11/730 RB730 Disk-based Systems.
   LESI refers to VAX-11/725 LESI/RC25 Disk-based System.
   UDA refers to a UDA/RA Disk-based System.

2) CRD considers this configuration an IDC Base System
   since it is assumed that the RB730 IDC will not be shipped
   as an option for CRD supported systems, and
   therefore, was originally shipped with the system.

1.3 Requirements

1.3.1 VAX-11/730 DUAL RL02 or RL02/R80 RB730 Disk-based Systems

1. System cabinet with:
   - MS730
   - CPU
   - 1MB Memory
   - one DMF32 Controller
   - one RB730 Disk Controller
   - RB730 Drive 1: RL02 Disk Drive
   - RB730 Drive 0: RL02 Disk Drive and RL02 Disk Cartridge, or R80 fixed Winchester disk drive
   - Dual TU58 Tape drives

2. LA120 Console Terminal
3. TU58#34 VAX-11/730 Console tape cartridge
4. TU58#36 VAX-11/730 Microdiagnostic tape cartridge
5. RL02#1 VAX-11/730 COMPLETE DIAGNOSTIC SET Disk Cartridge

1.3.2 VAX-11/730 TSU05 TS05/R80 RB730 Disk-based Systems

1. System cabinet with:
   - MS730
   - CPU
   - 1MB Memory
   - one RB730 Disk Controller
   - RB730 Drive 0: RL02 Disk Drive and RL02 Disk Cartridge
   - Dual TU58 Tape drives

2. VT52 Console Terminal
3. TU58#34 VAX-11/730 Console tape cartridge
4. TU58#36 VAX-11/730 Microdiagnostic tape cartridge
5. MT#1 VAX-11/730 COMPLETE DIAGNOSTIC SET Disk Cartridge

1.3.3 VAX-11/730 UDA/RA Disk-based System

1. System Cabinet with:
   - MS730
   - CPU
     1MB Memory
   - one UDA Disk Controller
     Dual TU58 Tape drives

. LA120 Console Terminal
3. TU58#34 VAX 11/730 Console tape cartridge
4. TU58#36 VAX 11/730 Microdiagnostic tape cartridge
5.
   For system with RA60 Drive 0 or 1:
   RA60#1 VAX-11/730 COMPLETE DIAGNOSTIC SET Disk Cartridge

   For system with no RA60 Drive 0 or 1 but Tape Subsystem:

MT#1 VAX-11/730 COMPLETE DIAGNOSTIC SET Tape

6. One of the following UDA Disk Drive configurations:
   A. RA60 Drive 1 and either RA60, 80, or 81 Drive 0
   B. RA60, 80 or 81 Drive 0

1.3.4 VAX-11/725 LESI/RC25 Disk-based System

1. System Cabinet with:
   - MS730
   - CPU
   - 1MB Memory
   - one LESI Disk Controller and RC25 Disk drive
   - Dual TU58 Tape drives
2. VT100 or LA120 Console Terminal
3. TU58#34 VAX-11/725 Console tape cartridge
4. TU58#36 VAX-11/725 Microdiagnostic tape cartridge
5. RC25#1 VAX-11/725 COMPLETE Diagnostic Set Disk Cartridge

2 CRD AUTO OFF-LINE

The AUTO Test mode of CRD verifies the operation of the CPU, the VMS system disk, and the diagnostic load disk for VAX-11/730,725. It 's invoked ONLY FROM VAX-11/730,725 CONSOLE by typing T to the Conso e prompt. Successful completion initiates CRD MENU Off ine.

2.1 Support

Base System	CRD AUTO Hardware Test Support (Standalone) (note 4)		
VAX-11/730 DUAL RL02	CPU	- KA730	
	FPA	- FP730	(note 7)
	MEMORY	- MS730	
	IDC	- RB730	
	COMBO	- DMF32	(note 3,7)
	Macro-Diagnostic Load Device	- IDC/RL02 DQA1 Drive 1	(note 1)
	System Disk	- IDC/RL02 DQA0 Drive 0	
VAX-11/730 TSU05 Disk	CPU	- KA730	
	FPA	- FP730	(note 7)
	MEMORY	- MS730	
	IDC	- RB730	
	COMBO	- DMF32	(note 3,7)
	Macro-Diagnostic Load Device	- IDC/TS05	
	System Device	- IDC/RL02 DQA1 Drive 0	
VAX-11/730 R80/RL02	CPU	- KA730	
	FPA	- FP730	(note 7)
	MEMORY	- MS730	
	IDC	- RB730	
	COMBO	- DMF32	(note 3,7)
	Macro-Diagnostic Load Device	- IDC/RL02 DQA1 Drive 1	(note 1)
	System Disk	- IDC/R80 DQA0 Drive 0	
VAX-11/725 LESI/RC25	CPU	- KA730	
	FPA	- FP730	(note 7)
	MEMORY	- MS730	
	Macro-Diagnostic Load Device	- LESI/RC25 DAA0 Drive 0 (removable)	(note 1,5)
	System Disk	LESI/RCF25	

		DAA1 Drive 1 (fixed)	
VAX-11/730 UDA/RA Disk based System	CPU	KA730	
	FPA	FP730	(note 7)
	MEMORY	MS730	
	COMBO	DMF32	(note 3,7)
	Macro-Diagnostic Load Device	UDA/RA60 DJA1 Drive 1	(note 2,6)
		Default: UDA/RA80,81,60 Drive 0	
	System Disk	UDA/RA81,80,60 Drive 0	

NOTES:

1) CRD will verify that the first seven(7) characters of the pack label are 'CRDPACK'.

2) If Drive 1 is an RA60 then Drive 1 will be used as load device; otherwise, Drive 0 will be used only if Drive 0 is either an RA81, RA80, or RA60 (Drive 1 RA60 takes precedence over Drive 0 RA60.)

   If the load device is an RA60 then CRD will verify that the first seven(7) characters of the pack label are 'CRDPACK'.

   In the default case, if Drive 0 is either an RA80 or RA81 then it is expected that all required CRD software is located on directory [SYSMAINT] or [SYS0.SYSMAINT] (note that in the default case the load device and system disk are the same).

3) Configured as lowest ranked in UNIBUS I/O Floating CSR Address space allocated for DMF32.

4) Unless otherwise specified, CRD AUTO will expect to find (available) and explicitly test the hardware specified for each VAX-11/730,725 Base System. Hardware not found (unavailable) will be considered a 'Fail' and 'incomplete' test condition.

5) LESI controller is configured at UNIBUS I/O Fixed address 772150.

6) UDA controller is configured at UNIBUS I/O Fixed address 772150.

7) Considered optional. If found (available), hardware is tested. If not found (unavailable), CRD specifies the hardware as 'unavailable', does not indicate a

'Fail' condition, and considers the testing 'complete'.

8) DMF32S and DMF32P ports considered optional. If found (available), hardware is tested. If not found (unavailable), CRD specifies the hardware as 'unavailable', does not indicate a 'Fail' condition, and considers the testing 'complete'. The DMF32A port is expected to be available. A 'Fail' and 'incomplete' test condition will result if unavailable.

9) Considered optional. If found (available), hardware is tested. If not found (unavailable), CRD continues.

2.2 Initiation Procedure

CRD AUTO Test provides automated testing of a VAX-11/730,725 packaged system. It does not test any additional devices beyond the base machine. To invoke the VAX-11/730,725 CRD AUTO Test, follow these steps:

1. CPU powered on with front panel 'key switch' to 'LOCAL' position and 'auto restart' switch to 'OFF' position. Insert TU58#34 VAX-11/730,725 Console tape cartridge in Drive 1. Insert TU58#36 VAX-11/730,725 Microdiagnostic tape cartridge in Drive 0

   NOTE

   Machine control must be in the hands of the Console program. This is indicated by the '>>> prompt at the console terminal. If the machine is powered off, then simply turn the keyswitch to the 'LOCAL' position. The machine will echo 'CONV___', then begin to load off the TU58#34 tape. The AUTO RESTART switch should be in the 'OFF' position, so that the machine halts once all microcode has been loaded. If the machine is running an operating system, use the recommended shutdown procedure to get back to the >>> Console prompt.

2. For VAX-11/730 DUAL RL02 or R80/RL02 Disk-based system:
   Insert RL02#1 VAX-11/730,725 BASIC DIAGNOSTIC SET Disk Cartridge into RL02 Drive 1. Make ready Drive 1 and Drive 0. Disks must be spinning with 'Write Protect' switch out (unlighted).

   WARNING

   EVRAD performs non-destructive disk testing by writing to the FE cylinder of an R80 drive, or writing to a duplicate copy of the 18 bit mode bad sector file which is not referenced by VAX/VMS for an RL02 drive. In the case of RL02 packs, the testing may be destructive if the pack is being used on an operating system other than VAX/VMS. If VAX/VMS is not the operating system being used on the machine, and the system drive is an RL02, then be sure to write-protect the drive or insert a scratch pack.

   For VAX-11/730 TSU05 TS05/RL02 Disk-based system:
   Contents of MT#1 BASIC DIAGNOSTIC SET must reside on Drive 0 account [SYSMAINT] or [SYS0.SYSMAINT]. Make ready Drive 0. Disks must be spinning with 'Write Protect' switch out (unlighted).

   WARNING

   TSU05 is a configuration which CRD cannot presently

distinguish from another configuration without manual operator intervention. The message which CRD Auto prints to properly determine the configuration is
**IF DRIVE 1 IS A TS05 THEN IGNORE ERROR MESSAGE **
** AND TYPE 2<CR> TO CONTINUE IF DRIVE 1 IS A **
** RL02 TYPE 1 <CR> TO CONTINUE TESTING **
Despite this message, the TS05 tape drive is tested only upon operator request in the CRD Menu section.

For VAX-11/730 UDA/RA Disk-based system:

- o If RA60 Drive 1:
  Insert RA60#1 VAX-11/730,725 BASIC DIAGNOSTIC SET Disk Cartridge into RA60 Drive 1. Make ready Drive 0 and 1. Disks must be spinning with 'Write Protect' switch out (unlighted).

- o No RA60 Drive 1:
  If Drive 0 RA60, insert RA60#1 VAX-11/730,725 BASIC DIAGNOSTIC SET Disk Cartridge into RA60 Drive 0.
  Otherwise, contents of RA60#1 BASIC DIAGNOSTIC SET must reside on Drive 0 account [SYSMAINT] or [SYS0.SYSMAINT]. Make ready Drive 0.

For VAX-11/725 LESI/RC25 Disk-based system:
Insert removable disk in drive. Push in 'RUN' switch. Push out Write Protect 'REMOVE' and 'FIXED' switch (unlighted). Wait for 'RUN' light to stop blinking.

WARNING

For release 15 only, CRD Auto will refer to the LESI/RC25 as a DUUA0. This is the (release 16) generic device name DAA0 needs further support.

3. 'T' Command
   Type T to the Console prompt and press RETURN
   Wait for CRD AUTO to start (30 seconds), and successfully complete (Approximately 15 minutes)

2.3 Program Flow - 'T' Console Command

CRD AUTO Test has the unique ability of executing the console program, the micro diagnostic monitor, a standalone level 4 diagnostic program, and the macro Diagnostic Supervisor, all in one continuous flow without operator intervention.

CRD begins with the machine under the control of the console program. The first action it takes in AUTO Test is to pass control to the CRD micro monitor. To do this, the console program, upon receiving a 'T' command from the terminal, searches the TU58 drives for the CRD micro monitor, ENKAB. When the file is located, it is loaded into WCS RAM, beginning at location 4C00. Unfortunately, parts of the console program and functional microcode also reside at these memory locations. This becomes very significant to CRD later, when control must be passed back to the console.

With the micro monitor loaded, the console program can simply execute a jump instruction to the starting address at 4C00(H) within the micro monitor. Control has been successfully passed, and all appropriate micro diagnostics can then be run.

Once the final micro has finished, given that no errors have occurred, CRD is ready to continue AUTO mode testing at the macro level. Control must be passed back to the console before testing resumes. The micro monitor can issue a jump back to the established console program address at 414D(H), and the code that was overwritten by MICMON must be reloaded into RAM. This is unfortunate from the point of view that reloading is a very time-consuming process, and it is being performed in an environment where time is very precious.

Once all the code has been reloaded, the console program will execute a command procedure CRAB00.CMD that will load and run the level 4 diagnostic, ENKAZ, then boot the Diagnostic Supervisor in CRD AUTO mode. The CRAB00.CMD command procedure will load both ENKAZ and VMB into main memory, then start ENKAZ. Following ENKAZ's Macro-Hardcore Instruction tests, it sizes the base disk drive configuration in order to perform a series of deposits to the general purpose registers which will tell VMB what the load device which the Supervisor is to run in AUTO mode on. The last instruction in ENKAZ is a jump to the starting address of the previously loaded VMB.

When machine control is in the hands of VMB while booting the Supervisor, CRD will lose the ability to display friendly error messages. VMB will not be changed to accommodate CRD, so standard error messages will be displayed if it fails during Supervisor boot. Therefore, the effort of CRD must be to minimize the possibility that VMB finds errors. This is why the IDC micro-diagnostic ENKCG and ENKAZ for the UDA must probe the diagnostic pack in search of the proper CRD pack label prior to booting the Supervisor.

Once the Diagnostic Supervisor is booted in CRD AUTO mode, it loads in its CRD control program, ENSAB. It then runs the Autosizer, EVSBA, to generate the proper ATTACH sequences for all devices in the packaged system. CRD AUTO Test then resumes testing at the macro diagnostic level.

Upon successful completion of AUTO Test macro diagnostics CRD will enter MENU mode. After the user has finished all menu testing, they will be able to exit back to the console program. For the first release of CRD, the only option available to the user was to exit since MENU testing had not been implemented at that time. This was accomplished by issuing a HALT instruction. Halting the main processor transfers machine control to the 8085-based console program.

2.4 Diagnostic Testing

2.4.1 VAX-11/730 DUAL RL02 or RL02/R80 RB730 Disk-based systems

CRD AUTO mode testing prints test titles for each hardware that it tests. Listed below are the test title names, as the customer sees them for a RB730 Disk-based system, and corresponding diagnostic tests executed.

TESTING	DIAGNOSTIC	TESTS	COMMENTS
CPU PART 1	ENKBA	1-4, 6-8	WCS RAM data test omitted.
	ENKBB	9-D, F	LS RAM data test omitted.
	ENKBC	ALL	
	ENKBD	ALL	
	ENKBE	ALL	
	ENKCA	ALL	
	ENKCB	ALL	
MEMORY	ENKCC	1-32	Moving inversions and UBE tests omitted.
CPU PART 2	ENKCD	ALL	
FPA	ENKCE	ALL	
	ENKCH	ALL	IDC Sizing Test, verifies if IDC is present. If it is not present micro-testing continues with ENKCF and ENKCG not run.
RB730	ENKCF	ALL	
RL02 PART 1	ENKCG	ALL	Only 2 tests available at this time. ENKCG is the IDC PART II Microdiagnostic. Its main purpose is to test the drivers and receivers of the IDC board which can only be tested with the drives attached. When run under CRD, ENKCG checks the Disk pack label of the RL02 pack verifying that the first seven characters are 'CRDPACK'.
CPU PART 3	ENKAZ	DEFAULT 1-5,6	1. Load VMB and ENKAZ. 2. Perform ENKAZ MACRO-HARDCORE INSTRUCTION TESTS. 3. Identify CRD Macro-diagnostic disk drive load path. 4. Load parameters into, then execute VMB which loads DIAGBOOT. 5. Execute DIAGBOOT which loads the Diagnostic Supervisor (VDS). 6. VDS loads and initiates CRD loadable 'mage.

TESTING	DIAGNOSTIC	TESTS	COMMENTS
			7. CRD types 'PASSED' to console indicating successful initiation of CRD Macro test.
	EVSBA	DEFAULT	Execute the Autosizer. Identify CRD base system type and system hardware. No test title is printed for this, neither is 'PASSED' of 'FAILED' printed upon completion.
R80 or RL02	EVRAD	DEFAULT	Quick flag set.
RL02 PART 2	EVRAD	DEFAULT	Quick flag set.
DMF32S	EVDLB	DEFAULT	Internal loopback.
DMF32A	EVDLC	DEFAULT	Internal loopback, Quick flag set.
DMF32P	EVDLD	DEFAULT	Internal loopback.

2.4.2 VAX-11/730 TSU05 TS05/R80 RB730 Disk-based systems

CRD AUTO mode testing prints test titles for each hardware that it tests. Listed below are the test title names, as the customer sees them for a RB730 Disk-based system, and corresponding diagnostic tests executed.

TESTING	DIAGNOSTIC	TESTS	COMMENTS
CPU PART 1	ENKBA	1-4, 6-8	WCS RAM data test omitted.
	ENKBB	9-D, F	LS RAM data test omitted.
	ENKBC	ALL	
	ENKBD	ALL	
	ENKBE	ALL	
	ENKCA	ALL	
	ENKCB	ALL	
MEMORY	ENKCC	1-32	Moving inversions and UBE tests omitted.
CPU PART 2	ENKCD	ALL	
FPA	ENKCE	ALL	
	ENKCH	ALL	IDC Sizing Test, verifies if IDC is present. If it is not present micro-testing continues with ENKCF and ENKCG not run.
RB730	ENKCF	ALL	
RL02 PART 1	ENKCG	ALL	Only 2 tests available at this time. ENKCG is the IDC PART II Microdiagnostic. Its main purpose is to test the drivers and receivers of the IDC board which can only be tested with the drives attached. When run under CRD, ENKCG checks the Disk pack label of the RL02 pack verifying that the first seven characters are 'CRDPACK'.
CPU PART 3	ENKAZ	DEFAULT 1-5,6	1. Load VMB and ENKAZ. 2. Perform ENKAZ MACRO-HARDCORE INSTRUCTION TESTS. 3. Identify CRD Macro-diagnostic disk drive load path. 4. Load parameters into, then execute VMB which loads DIAGBOOT. 5. Execute DIAGBOOT which loads the Diagnostic Supervisor (VDS). 6. VDS loads and initiates CRD loadable image.

TESTING	DIAGNOSTIC	TESTS	COMMENTS
			7. CRD types 'PASSED' to console indicating successful initiation of CRD Macro test.
	EVSBA	DEFAULT	Execute the Autosizer. Identify CRD base system type and system hardware. No test title is printed for this, neither is 'PASSED' of 'FAILED' printed upon completion.
RL02 PART 2	EVRAD	DEFAULT	Quick flag set.
DMF32S	EVDLB	DEFAULT	Internal loopback.
DMF32A	EVDLC	DEFAULT	Internal loopback, Quick flag set.
DMF32P	EVDLD	DEFAULT	Internal loopback.

2.4.3 VAX-11/730 UDA/RA Disk-based systems

CRD AUTO mode testing prints test titles for each hardware that it tests. Listed below are the test title names, as the customer sees them for a UDA/RA Disk-based system, and corresponding diagnostic tests executed.

TESTING	DIAGNOSTIC	TESTS	COMMENTS
CPU PART 1	ENKBA	1-4, 6-8	WCS RAM data test omitted.
	ENKBB	9-D, F	LS RAM data test omitted.
	ENKBC	ALL	
	ENKBD	ALL	
	ENKBE	ALL	
	ENKCA	ALL	
	ENKCB	ALL	
MEMORY	ENKCC	1-32	Moving inversions and UBE tests omitted.
CPU PART 2	ENKCD	ALL	
FPA	ENKCE	ALL	
	ENKCH	ALL	IDC Sizing Test, verifies if IDC is present. If it is not present micro-testing continues to CPU PART 3 testing.
CPU PART 3	ENKAZ	DEFAULT 1 5,6,7	1. Load VMB and ENKAZ. 2. Perform ENKAZ MACRO-HARDCORE INSTRUCTION TESTS. 3. Identify CRD Macro-diagnostic disk drive load path. Verify that pack is spinning. Print 'PASSED' if no errors.
RA60 PART 1 RA80 PART 1 RA81 PART 1	ENKAZ	8	1. If Macro-diagnostic disk pack is RA60, then verify RA60 pack label 'CRDPACK'. Test 8 ENKAZ. 2. Load parameters into, then execute VMB which loads DIAGBOOT. 3. Execute DIAGBOOT which loads the Diagnostic Supervisor (VDS). 4. VDS loads and initiates CRD loadable image. 5. CRD types 'PASSED' to console indicating successful initiation of CRD Macro test.

TESTING	DIAGNOSTIC	TESTS	COMMENTS
	EVSBA	DEFAULT	Execute the Autosizer. Identify CRD base system type and system hardware. No test title is printed here, neither is 'PASSED' or 'FAILED' printed.
RA81 RA80 RA60	EVRLA	1,2	Quick flag set. Section 'CRD12'
RA81 PART 2 RA80 PART 2 RA60 PART 2	EVRLA	1,2	Quick flag set. Section 'CRD12'
DMF32S	EVDLB	DEFAULT	Internal loopback.
DMF32A	EVDLC	DEFAULT	Internal loopback, Quick flag set.
DMF32P	EVDLD	DEFAULT	Internal loopback.

2.4.4 VAX-11/725 LESI/RC25 Disk-based systems

CRD AUTO mode testing prints test titles for each hardware that it tests. Listed below are the test title names, as the customer sees them for a LESI/RC25 Disk-based system, and corresponding diagnostic tests executed.

TESTING	DIAGNOSTIC	TESTS	COMMENTS
CPU PART 1	ENKBA	1-4, 6-8	WCS RAM data test omitted.
	ENKBB	9-D, F	LS RAM data test omitted.
	ENKBC	ALL	
	ENKBD	ALL	
	ENKBE	ALL	
	ENKCA	ALL	
	ENKCB	ALL	
MEMORY	ENKCC	1-32	Moving inversions and UBE tests omitted.
CPU PART 2	ENKCD	ALL	
FPA	ENKCE	ALL	
	ENKCH	ALL	IDC Sizing Test, verifies if IDC is present. If it is not present micro-testing continues with ENKCF and ENKCG not run.
CPU PART 3	ENKAZ	DEFAULT 1-5,6	1. Load VMB and ENKAZ. 2. Perform ENKAZ MACRO-HARDCORE INSTRUCTION TESTS. 3. Identify CRD Macro-diagnostic disk drive load path. Print 'PASSED' if no error.
RC25 PART 1	ENKAZ	8	1. Verify Pack Lable 'CRDPACK'. 2. Load parameters into, then execute VMB which loads DIAGBOOT. 3. Execute DIAGBOOT which loads the Diagnostic Supervisor (VDS). 4. VDS loads and initiates CRD loadable image. 5. CRD types 'PASSED' to console indicating successful initiation of CRD Macro test.
	EVSBA	DEFAULT	Execute the Autosizer. Identify CRD base system type and system hardware. No test title is printed for this, neither is 'PASSED' of 'FAILED' printed upon completion.
RC25 PART 2	EVRMB		Quick flag set.

TESTING	DIAGNOSTIC	TESTS	COMMENTS
RCF25	EVRMB		Quick flag set.

2.5 Sample Printouts

The following pages detail what may be displayed on the console terminal as CRD AUTO Test progresses.

NOTE

- The format and content of these printouts will change as CRD evolves.

The run times listed with each test are rounded to the nearest appropriate 15 second interval. They should be used as a guide to indicate the approximate end of the testing. Memory testing is based on the standard 1 Megabyte of memory. Each additional Megabyte of memory will add about 20 seconds to the program's run time. If testing continues well beyond the expected run time, it should be assumed that CRD encountered an unrecoverable error. Recheck the setup requirements listed above, and call Digital Field Service if the problem remains.

2.5.1 Printouts for VAX-11/730 UDA/RA Disk-based Systems

2.5.1.1 Printout for a UDA kernel system.

The following is an output of a successful CRD AUTO Test run for a UDA kernel system. CRD will automatically size the system and execute diagnostics accordingly.

```
>>> T

****    BEGIN VAX-11/730,725 AUTO TEST      ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES

CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED
MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED
CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED
FPA                 TESTING STARTED - RUN TIME = 1:45    PASSED
CPU PART 3          TESTING STARTED - RUN TIME = 4:00    PASSED
RA60 PART 1   DJA1  TESTING STARTED - RUN TIME = 0:30    PASSED
RA80          DUA0  TESTING STARTED - RUN TIME = 2:45    PASSED
RA60 PART 2   DJA1  TESTING STARTED - RUN TIME = 1:00    PASSED
DMF32S        XGA0  TESTING STARTED - RUN TIME = 1:00    PASSED
DMF32A   TXA0-TXA7  TESTING STARTED - RUN TIME = 0:30    PASSED
DMF32P        LCA0  TESTING STARTED - RUN TIME = 0:15    PASSED

AUTO TEST COMPLETE - NO ERRORS
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                        VERSION 1.5

**** BEGIN VAX-11/730,725 MENU TEST ****

( ... You are now in menu mode, see MENU sample printouts ... )
```

2.5.2 Printouts for VAX-11/725 LESI/RC25 Disk-based Systems

2.5.2.1 Printout for a LESI/RC25 Disk-based system.

The following is an output of a successful CRD AUTO Test run for a LESI/RC25 Disk-based system. CRD will automatically size the system and execute diagnostics accordingly.

```
>>> T

****    BEGIN VAX-11/730,725 AUTO TEST      ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES

CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED
MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED
CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED
FPA                 TESTING STARTED - RUN TIME = 1:45    PASSED
CPU PART 3          TESTING STARTED - RUN TIME = 4:00    PASSED
RC25 PART 1   DAA0  TESTING STARTED - RUN TIME = 0:30    PASSED
RC25 PART 2   DAA0  TESTING STARTED - RUN TIME = 1:00    PASSED
RCF25         DAA1  TESTING STARTED - RUN TIME = 1:00    PASSED

AUTO TEST COMPLETE - NO ERRORS
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                        VERSION 1.5

**** BEGIN VAX-11/730,725 MENU TEST ****

( ... You are now in menu mode, see MENU sample printouts ... )
```

2.5.3 Printouts for VAX-11/730 RL02/R80 RB730 Disk-based Systems

2.5.3.1 Printout for system without the FPA option

The following is an output of a successful CRD AUTO Test run for a system without the FPA option. CRD will automatically size the system for this option, and execute the diagnostic accordingly.

```
>>> T

****    BEGIN VAX-11/730,725 AUTO TEST      ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES

CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED

MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED

CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED

FPA                 TESTING STARTED - RUN TIME = 1:45
FPA NOT AVAILABLE

RB730               TESTING STARTED - RUN TIME = 0:45    PASSED

RL02 PART 1   DQA1  TESTING STARTED - RUN TIME = 0:15    PASSED

CPU PART 3          TESTING STARTED - RUN TIME = 4:00    PASSED

R80           DQA0  TESTING STARTED - RUN TIME = 0:30    PASSED

RL02 PART 2   DQA1  TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32S        XGA0  TESTING STARTED - RUN TIME = 1:00    PASSED

DMF32A   TXA0-TXA7  TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32P        LCA0  TESTING STARTED - RUN TIME = 0:15    PASSED

AUTO TEST COMPLETE - NO ERRORS
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                        VERSION 1.5

**** BEGIN VAX-11/730,725 MENU TEST ****

( ... You are now in menu mode, see MENU sample printouts ... )
```

Situations can arise that prevent complete CRD testing. Probably the most common of these special cases is when a given device in the Packaged System cannot be found by the diagnostics. For example, if a disk drive is powered off or disconnected, it will not be seen. Or, if the switchpacks on the DMF32 (COMBO) board are not set for the proper UNIBUS address, it may not be found. A similar situation arises if any of the DMF32 ports are disabled via the switchpack on the distribution panel. In all of these cases, a message will notify the user that the device is unavailable. If possible, CRD will resume testing of the next device. CRD cannot continue without the presence of all three CPU boards. It also requires the presence of the RB730 in order to perform disk and DMF32 testing.

2.5.3.2 Printout for system where DMF32P and DMF32A are disabled

The following is a printout where both the synchronous and parallel ports of the DMF32 board have been disabled:

```
>>> T

****    BEGIN VAX-11/730,725 AUTO TEST      ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES

CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED

MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED

CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED

FPA                 TESTING STARTED - RUN TIME = 1:45    PASSED

RB730               TESTING STARTED - RUN TIME = 0:45    PASSED

RL02 PART 1   DQA1  TESTING STARTED - RUN TIME = 0:15    PASSED

CPU PART 3          TESTING STARTED - RUN TIME = 4:00    PASSED

R80           DQA0  TESTING STARTED - RUN TIME = 0:30    PASSED

RL02 PART 2   DQA1  TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32S        XGA0  TESTING STARTED - RUN TIME = 1:00
DEVICE NOT AVAILABLE

DMF32A   TXA0-TXA7  TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32P        LCA0  TESTING STARTED - RUN TIME = 0:15
DEVICE NOT AVAILABLE


AUTO TEST COMPLETE - NO ERRORS
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                        VERSION 1.5

**** BEGIN VAX 11/730,725 MENU TEST ****

( ... You are now in menu mode, see MENU sample printouts ... )
```

For CRD AUTO Test to run properly, the correct setup described earlier must be performed. The three required media are the microcode and diagnostic TU58 tapes, and the diagnostic RL02 pack.

If the system microcode tape does not support CRD, then the 'T command issued to begin testing will first cause files to be loaded from the TU58 tapes and will then result in a MIC> prompt displayed at the console terminal. If this should happen, a RETURN command given to the MIC> prompt will reload the Console program.

If the TU58 tape inserted in drive 0 does not contain the CRD AUTO Test control program, the following message will be printed, and control will remain in the Console program:

2.5.3.3 Printout for system where TU58 in drive 0 does not contain CRD AUTO Test

```
>>> T

?40  CRD TU-58 NOT FOUND

>>>
```

2.5.3.4 Printout for system where TU58 in drive 0 has missing or corrupt files

If the diagnostic tape in TU58 drive 0 has files missing or corrupted, CRD cannot continue and testing is aborted. The following is a printout of a CRD AUTO Test run where a problem with the diagnostic TU58 tape arises:

```
>>> T

****    BEGIN VAX-11/730,725 AUTO TEST      ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES


CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED

MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED

CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED

**     PROPER SETUP REQUIRED - CANNOT CONTINUE      **
**  CHECK DIAGNOSTIC TAPE INSERTED IN TU58 DRIVE 0  **
** IF SETUP CORRECT, POSSIBLE TAPE OR DRIVE PROBLEM **

AUTO TEST ABORTED
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE
                        VERSION 1.5
**** BEGIN VAX 11/730,725 MENU TEST ****
( ... You are now in menu mode, see MENU sample printouts ... )
```

The correct RL02 disk pack must be up and spinning in drive DQA1: for CRD to execute properly. To insure this, a quick test is made to verify that DQA1: contains the RL02 pack labeled 'CRDPACK . If this is not the case, a message is displayed asking the user to set up the drive. Once this has been completed, the user types a carriage return to restart testing. This is the only possibility of manual intervention during CRD AUTO Test.

2.5.3.5 Printout for system where user forgot to spin up RL02 pack

The following is a printout of a successful CRD AUTO Test run, where the user has forgotten to spin up the diagnostic RL02 pack:

```
>>> T

****     BEGIN VAX-11/730,725 AUTO TEST     ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES


CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED

MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED

CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED

FPA                 TESTING STARTED - RUN TIME = 1:45    PASSED

RB730               TESTING STARTED - RUN TIME = 0:45    PASSED

RL02 PART 1   DQA1 TESTING STARTED - RUN TIME = 0:15

** PROPER SETUP REQUIRED - TYPE <CR> TO CONTINUE **
**   CHECK FOR RL02 DIAGNOSTIC PACK 'CRDPACK'     **
**     INSERTED AND SPINNING IN DRIVE DQA1:       **
** IF SETUP CORRECT, POSSIBLE RL02 DRIVE PROBLEM **

CPU PART 3          TESTING STARTED - RUN TIME = 4:00    PASSED

R80           DQA0 TESTING STARTED - RUN TIME = 0:30    PASSED

RL02 PART 2   DQA1 TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32S        XGA0 TESTING STARTED - RUN TIME = 1:00    PASSED

DMF32A   TXA0-TXA7 TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32P        LCA0 TESTING STARTED - RUN TIME = 0:15    PASSED

AUTO TEST COMPLETE - NO ERRORS
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                        VERSION 1.5

**** BEGIN VAX-11/730,725 MENU TEST ****

( ... You are now in menu mode, see MENU sample printouts ... )
```

Two examples of incomplete disk drive testing are when the drives are write-protected or not spun up. If either of the cases are found under CRD, messages will be displayed on the console terminal, and testing will resume with the next device.

2.5.3.6 Printout for system where system pack has been write-protected

The following is a printout of a successful CRD AUTO Test run, but where the system pack, DQA0:, has been write-protected.

```
>>> T

****     BEGIN VAX-11/730,725 AUTO TEST     ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES

CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED
MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED
CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED
FPA                 TESTING STARTED - RUN TIME = 1:45    PASSED
RB730               TESTING STARTED - RUN TIME = 0:45    PASSED
RL02 PART 1   DQA1  TESTING STARTED - RUN TIME = 0:15    PASSED
CPU PART 3          TESTING STARTED - RUN TIME = 4:00    PASSED
R80           DQA0  TESTING STARTED - RUN TIME = 0:30

**       UNIT IS WRITE-PROTECTED        **
** DQA0: DISK DRIVE TESTING INCOMPLETE  **

RL02 PART 2   DQA1  TESTING STARTED - RUN TIME = 0:30    PASSED
DMF32S        XGA0  TESTING STARTED - RUN TIME = 1:00    PASSED
DMF32A   TXA0-TXA7  TESTING STARTED - RUN TIME = 0:30    PASSED
DMF32P        LCA0  TESTING STARTED - RUN TIME = 0:15    PASSED

AUTO TEST INCOMPLETE - NO ERRORS
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                        VERSION 1.5

**** BEGIN VAX-11/730,725 MENU TEST ****

( ... You are now in menu mode, see MENU sample printouts ... )
```

2.5.3.7 Printout for system where system pack is not spun up

The following is an output of a successful CRD AUTO Test run where the system disk, DQA0:, is not spun up.

```
>>> T

****     BEGIN VAX-11/730,725 AUTO TEST     ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES

CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED

MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED

CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED

FPA                 TESTING STARTED - RUN TIME = 1:45    PASSED

RB730               TESTING STARTED - RUN TIME = 0:45    PASSED

RL02 PART 1   DQA1 TESTING STARTED - RUN TIME = 0:15    PASSED

CPU PART 3          TESTING STARTED - RUN TIME = 4:00    PASSED

R80           DQA0 TESTING STARTED - RUN TIME = 0:30

** DRIVE NOT ONLINE - CHECK DRIVE SPUN UP **
**   DQA0: DISK DRIVE TESTING INCOMPLETE  **

RL02 PART 2   DQA1 TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32S        XGA0 TESTING STARTED - RUN TIME = 1:00    PASSED

DMF32A   TXA0-TXA7 TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32P        LCA0 TESTING STARTED - RUN TIME = 0:15    PASSED


AUTO TEST INCOMPLETE -
CHECK DEVICE PREPARATION - IF OK, THEN CALL FIELD SERVICE
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                        VERSION 1.5

**** BEGIN VAX-11/730,725 MENU TEST ****

( ... You are now in menu mode, see MENU samp e printouts ... )
```

Any hard error found by the diagnostics while running CRD AUTO Test will cause an immediate failure message to be displayed. Errors will always cause testing of the device to be aborted. However, in some cases CRD AUTO Test will abort, and in others it will resume testing of the next device. When CRD testing must be aborted (in the case of CPU, Memory, FPA, and RB730 errors), the user will be offered the choice of returning back to the Console program or restarting AUTO Test. When testing can continue (in the case of

disk or DMF32 errors), CRD AUTO will run to completion and return to the Console program. Note that CRD MENU is not invoked when CRD AUTO detects a hard error.

The following are printouts of CRD AUTO Test runs with errors

found in various diagnostics.

2.5.3.8 Printout for system where error in RB730 testing

The next printout is displayed when an error is encountered during the execution of RB730 testing. Testing cannot continue from this error. Note that in this example the user typed a 1 in response to the ENTER 1 TO RETURN TO CONSOLE, 2 TO RESTART AUTO TEST prompt.

```
>>> T

****     BEGIN VAX-11/730,725 AUTO TEST     ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES

CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED

MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED

CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED

FPA                 TESTING STARTED - RUN TIME = 1:45    PASSED

RB730               TESTING STARTED - RUN TIME = 0:45   *FAILED*

AUTO TEST ABORTED - RB730 BAD - CALL FIELD SERVICE

ENTER 1 TO RETURN TO CONSOLE, 2 TO RESTART AUTO TEST 1
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                        VERSION 1.5

**** BEGIN VAX-11/730,725 MENU TEST ****

( ... You are now in menu mode, see MENU sample printouts ... )
```

2.5.3.9 Printout for system where error in DMF32 testing

The following printout is displayed when an error is encountered during DMF32 synchronous port testing. Testing resumes with DMF32 asynchronous port testing.

```
>>> T

****     BEGIN VAX-11/730,725 AUTO TEST     ****
VERSION 1.5 RUN TIME APPROXIMATELY 15:00 MINUTES

CPU PART 1          TESTING STARTED - RUN TIME = 3:30    PASSED

MEMORY              TESTING STARTED - RUN TIME = 1:30    PASSED

CPU PART 2          TESTING STARTED - RUN TIME = 0:45    PASSED

FPA                 TESTING STARTED - RUN TIME = 1:45
FPA NOT AVAILABLE

RB730               TESTING STARTED - RUN TIME = 0:45    PASSED

RL02 PART 1   DQA1  TESTING STARTED - RUN TIME = 0:15    PASSED

CPU PART 3          TESTING STARTED - RUN TIME = 4:00    PASSED

R80           DQA0  TESTING STARTED - RUN TIME = 0:30    PASSED

RL02 PART 2   DQA1  TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32S        XGA0  TESTING STARTED - RUN TIME = 1:00   *FAILED*

DMF32A   TXA0-TXA7  TESTING STARTED - RUN TIME = 0:30    PASSED

DMF32P        LCA0  TESTING STARTED - RUN TIME = 0:15    PASSED

AUTO TEST COMPLETE - XGA0 BAD - CALL FIELD SERVICE
**** END OF VAX-11/730,725 AUTO TEST ****

                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                          VERSION 1.5

**** BEGIN VAX-11/730,725 MENU TEST ****

( ... You are now in menu mode, see MENU sample printouts ... )
```

3 CRD MENU OFF-LINE

The Menu-driven test mode of CRD verifies the operation of all subsystems including options for VAX-11/730,725. The Menu format provides a simple interface to select and test either one or all VAX subsytems. It is invoked by any one of the following three methods:

- typing T/M" to VAX-11/730,725 Console prompt,
- typing 'CRD to VAX Diagnostic Supervisor prompt, or
- typing T' to VAX-11/730,725 Console prompt ('T invokes CRD AUTO which then invokes CRD MENU).

NOTE

CRD MENU is not supported for use in Diagnostic Supervisor command files. The results of having a 'CRD' command in a Supervisor command file are unpredictab e.

3.1 Support - Hardware/Diagnostic

The following hardware is explicitly tested in CRD MENU Off-line by the VAX Diagnostic files specified:

HARDWARE	DIAGNOSTIC FILES REQUIRED
KA730 CPU	ENKAX, EVKAB, EVKAC, EVKAD, EVKAE
RB730/RL02,R80	EVRAD, EVQDQ (driver)
RK711	EVREA, EVREB, EVREC, EVRED, EVREE
RK711/RK07	EVRAD, EVQDM (driver)
RL211/RL02	EVRAD, EVQDL (driver)
UDA/RA80,81,60	EVRLA, EVRLA.DAT (Data File)
DMF32S	EVDLB
DMF32A	EVDLC
DMF32P	EVDLD
DR11W	EVDRE
DUP11	EVDUP
DZ32	EVDAB
TS11	EVMAD, EVQTS (driver)
TU80	EVMBE
TS05	EVMAI
RC25	EVRMB
VS100	EVWAB
DEUNA	EVDWA

3.2 Initiation Procedure

There are three ways to invoke CRD MENU Off-line:

- o by T/M Console Command,
- o by CRD VAX Diagnostic Supervisor command,
- o by T Console command (T invokes CRD AUTO which then invokes CRD MENU)

Note that the times quoted below are approximations, and depend on the size of the system, which version of CRD is running, and various other factors.

1. CPU powered on with front panel 'key switch' to 'LOCAL' position and 'auto restart' switch to 'OFF' position
2. Insert TU58#34 VAX-11/730,725 Console tape cartridge in Drive 1
3. Insert TU58#36 VAX-11/730,725 Microdiagnostic tape cartridge in Drive 0
4.

   For VAX-11/730 DUAL RL02 or R80/RL02 Disk-based system:
   Insert RL02#1 VAX-11/730,725 BASIC DIAGNOSTIC SET Disk Cartridge into RL02 Drive 1. Make ready Drive 1 and Drive 0.

   For VAX-11/730 TSU05 TS05/RL02 Disk-based system:

   Contents of MT#1 BASIC DIAGNOSTIC SET must reside on Drive 0 account [SYSMAINT] or [SYS0.SYSMAINT]. Make ready Drive 0.

   For VAX-11/730 UDA/RA Disk-based system:
   - o If RA60 Drive 1:
     Insert RA60#1 VAX-11/730,725 BASIC DIAGNOSTIC SET Disk Cartridge into RA60 Drive 1. Make ready Drive 0 and 1.
   - o No RA60 Drive 1:
     If Drive 0 RA60, insert RA60#1 VAX-11/730,725 BASIC DIAGNOSTIC SET Disk Cartridge into RA60 Drive 0.
     Otherwise, contents of RA60#1 BASIC DIAGNOSTIC SET must reside on Drive 0 account [SYSMAINT] or [SYS0.SYSMAINT]. Make ready Drive 0.

   For VAX-11/725 LESI/RC25 Disk-based system:
   Insert removable disk in drive. Push in 'RUN' switch. Push out Write Protect 'REMOVE' and 'FIXED' switch (unlighted). Wait for 'RUN' light to stop blinking.

5.

   'T' Command

- Type T to the Console prompt and press RETURN
- Wait for CRD AUTOTEST to start (30 seconds), and successfully complete (Approximately 15 minutes)
- Wait for CRD MENU to print Main Menu.

'T/M' Command
- Type T/M to the Console prompt and press RETURN
- Wait for CRD Off-Line Main Menu to be printed (response time 1:30 minutes).

'CRD' Command
- Boot Diagnostic Supervisor by typing to the console 'B SQ1' for IDC based system, or 'B SUx' if UDA based system or 'B SA0' if LESI based system. Press RETURN .
- Wait for Supervisor prompt (1:45 minutes). Type CRD and press RETURN
  Wait for CRD Off-line Main Menu to be printed.

3.3 Diagnostic Testing

A hardware test title is printed each time CRD MENU starts testing a hardware device selected for test. For each hardware test title the following shows both the VAX Diagnostic Supervisor commands used by CRD MENU to start testing and a description of testing being performed:

KA730 PART 1 KA0

- VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load ENKAX
  - Set Flags Quick
  - Select KA0
  - Start
- Testing:
  - KA730 specific CPU cluster exercising

KA730 PART 2 KA0

- VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVKAB
  - Set Flags Quick
  - Select KA0
  - Start
- Testing:
  - Basic CPU Instruction Exercising

KA730 PART 3 KA0

- VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVKAC
  - Set Flags Quick
  - Select KA0
  - Start
- Testing:
  - Floating Point Testing

KA730 PART 4 KA0

- o VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVKAD
  - Set Flags Quick
  - Select KA0
  - Start
- o Testing:
  - CPU Compatibility Mode Testing

KA730 PART 5 KA0

- o VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVKAE
  - Set Flags Quick
  - Select KA0
  - Start
- o Testing:
  - CPU Privileged Architecture testing

RK711 PART 1 'Generic Device Name'

- o VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVREA
  - Set Flags Quick
  - Select 'Generic Device Name'
  - Start

RK711 PART 2 'Generic Device Name'

- o VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVREB
  - Set Flags Quick
  - Select 'Generic Device Name'
  - Start

RK711 PART 3 'Generic Device Name'

- o VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVREC
  Set Flags Quick
  Select 'Generic Device Name'
  Start

RK711 PART 4 'Generic Device Name'

- o VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVRED
  Set Flags Quick
  Se ect 'Generic Device Name'
  Start

RK711 PART 5 'Generic Device Name'

- o VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVREE
  Set Flags Quick
  Select 'Generic Device Name'
  Start

DMF32S COMBO 'Generic Device Name'

- o VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVDLB
  Set Flags Quick
  Select 'Generic Device Name'
  Start

- o Testing:
  Internal wrap testing: EIA drivers, receivers or distribution pane not tested

DMF32A COMBO 'Generic Device Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVDLC
  Set Flags Quick
  Select 'Generic Device Name'
  Start
- Testing:
  Internal Wrap testing

DMF32P COMBO 'Generic Device Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVDLD
  Set Flags Quick
  Select 'Generic Device Name'
  Start
- Testing:
  If printer is attached then Registers and LP logic tested;
  otherwise only Registers tested

RL02 DISK 'Generic Device Name'
R80 DISK 'Generic Device Name'
RK07 DISK 'Generic Device Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVRAD
  Set Flags Quick
  Select 'Generic device name'
  Start
- Testing:
  Functional reads and writes; non-destructive testing. Disks with useable data may be used.

RA80 DISK 'Generic Device Name'
RA81 DISK 'Generic Device Name'
RA60 DISK 'Generic Device Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVRLA
  Set Flags Quick
  Select 'Generic device name'
  Start/Section:CRD12

- Testing:
  Tests 1 and 2.

RCF25 DISK 'Generic Device Name'
RC25 DISK 'Generic Device Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVRMB
  Set Flags Quick
  Select 'Generic device name'
  Start/Section:CRD

TS11 TAPE 'Generic Device Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVMAD
  Set Flags Quick
  Select 'Generic Device Name'
  Start

TS05 TAPE 'Generic Dev'ce Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVMAI
  Set Flags Quick
  Select 'Generic Device Name'
  Start/Secton:CRD

TU80 TAPE 'Generic Device Name'

- VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVMBE
  - Set Flags Quick
  - Select 'Generic Device Name'
  - Start/Secton:CRD

DR11W REALTIME 'Generic Device Name'

- VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVDRE
  - Set Flags Quick
  - Select 'Generic Device Name'
  - Start
- Testing:
  - Internal wrap testing

DUP11 COMM. 'Generic Device Name'

- VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVDUP
  - Set Flags Quick
  - Select 'Generic Device Name'
  - Start
- Testing:
  - Internal wrap testing

DZ32 COMM. 'Generic Device Name'

- VDS Commands:
  - Clear Flags All
  - Clear Event Flags All
  - Load EVDAB
  - Set Flags Quick
  - Select 'Gener'c Device Name'
  - Start
- Testing:
  - Internal wrap test'ng

DUP11 COMM. 'Generic Device Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVDUP
  Set Flags Quick
  Select 'Generic Device Name'
  Start

- Testing:
  Internal wrap testing

VS100 COMM. 'Generic Device Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVWAB
  Set Flags Quick
  Select 'Generic Device Name'
  Start

DEUNA COMM. 'Generic Device Name'

- VDS Commands:
  Clear Flags All
  Clear Event Flags All
  Load EVDWA
  Set Flags Quick
  Select 'Generic Device Name'
  Start

3.4 Sample Printouts

The following printouts show various stages of CRD MENU Test. All of these assume CRD MENU was invoked from the T/M console command or from a successful completion of CRD AUTO Test.

NOTE

The format and content of these printouts will change as CRD evolves. The times given in the following printouts are approximations only.

3.4.1 MAIN MENU (Functional Test) Printout

The following is printed as a result of invoking VAX-11/730,725 CRD MENU Test.

```
                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                          VERSION 1.5


**** BEGIN VAX-11/730,725 MENU TEST ****

Type the CTRL key and the C key (together) at
any time to interrupt Menu Test Processing


VAX-11/730,725 HARDWARE IDENTIFICATION  - RUN TIME = 00:30 Minutes
PLEASE WAIT . . .

VAX-11/730,725 HARDWARE IDENTIFICATION COMPLETE


---------------------------
 MAIN MENU - Functional Test
---------------------- ----

   A =  Exit Menu Test
   B =  Print list of identified system hardware and support status
   C =  Print preparation required for test of supported hardware
   D =  Select and Test hardware from TEST MENU
   E =  TEST ALL Supported Hardware

   Type one of the above (e.g. A), and press RETURN,
   Enter MAIN MENU choice:
```

3.4.1.1 Exit Menu Test Printout

The following is printed as a result of choosing Exit Menu Test from the Main Menu.

```
    ( from Main Menu )
            .
            .
            .
Type one of the above (e.g. A), and press RETURN,

Enter MAIN MENU choice: A

MENU TEST STOPPED BY OPERATOR
**** END OF VAX-11/730,725 MENU TEST ****

RETURNING TO VAX-11/730,725 CONSOLE, WAIT FOR  >>>  PROMPT

?02 PC=00000000
>>>
```

3.4.1.2 Print System Hardware Printout

The following is printed as a result of choosing Print list of identified system hardware . . . from the Main Menu.

(from Main Menu)

.
.
.

Type one of the above (e.g. A), and press RETURN

Enter MAIN MENU choice: B

Hardware Name	Generic Name	Test Menu Support
console	CSA0	No Support
DW730	DW0	No Support
RB730	DQA	No Support
RL02	DQA1	SUPPORTED
KA730	KA0	SUPPORTED
RK711	DMA	SUPPORTED
RK07	DMA0	SUPPORTED
RL11	DLA	No Support
RL02	DLA0	SUPPORTED
R80	DQA0	SUPPORTED
DMF32S	XGA0	SUPPORTED
DMF32A	TXA	SUPPORTED
DMF32P	LCA	SUPPORTED

.
.
.

(return to Main Menu)

3.4.1.3 Print Hardware Preparation Printout

The following is printed as a result of choosing Print hardware preparation . . . from the Main Menu.

```
        ( from Main Menu )
                .
                .
                .
    Type one of the above (e.g. A), and press RETURN,

    Enter MAIN MENU choice: C


   Hardware:    DMF32A, DMF32P, DMF32S, KA730, RK711
Preparation:    1. No Hardware Preparation required.


   Hardware:    RL02, RK07
Preparation:    1. Insert disk cartridge in drive.  Disk may contain
                   useable data.
                2. Push In  LOAD  switch.
                3. Push out  WRITE PROT  switch (not lighted).
                4. Wait for  READY  indicator to light.


   Hardware:    R80
Preparation:    1. Push in  RUN STOP  switch.
                2. Push out  WRITE PROT  switch (not lighted).
                3. Wait for  READY  indicator to light.

                  .
                  .
                  .
        ( return to Main Menu )
```

3.4.2 TEST MENU Printout

The following is the menu to select either one or all supported hardware for testing. This menu is entered as a result of choosing Select and test one or all suppported hardware from the Main Menu.

(from Main Menu)

.
.
.

Type one of the above (e.g. A), and press RETURN,

Enter MAIN MENU choice: D

```
------------
 TEST MENU
-----------

  A1 =  KA730   CPU       KA0

  B1 =  DMF32S  COMM.     XGA0
  B2 =  DMF32A  COMM.     TXA
  B3 =  DMF32P  COMM.     LCA

  C1 =  RL02    DISK      DLA0
  C2 =  R80     DISK      DQA0
  C3 =  RL02    DISK      DQA1

  D1 =  TEST ALL of the above supported hardware

  Type one of the above (e.g. A1), and press RETURN to start testing.

  Enter TEST MENU choice:
```

3.4.2.1 TEST ALL - No Errors Printout

The following is a printout of a successful CRD MENU Functional test run, as a result of choosing TEST ALL from the Test Menu.

```
        ( from Main Menu )
                .
                .
                .
   Type one of the above (e.g. A1), and press RETURN to start testing.

   Enter TEST MENU choice: D1


    Type the CTRL key and the C key (together)
    at any time to interrupt Functional Test...


    ********** BEGIN FUNCTIONAL TESTING **********

               RUN TIME = 15:00 MINUTES

KA730   PART 1     KA0    TESTING STARTED - RUN TIME = 00:45 MINUTES   PASSED
KA730   PART 2     KA0    TESTING STARTED - RUN TIME = 02:00 MINUTES   PASSED
KA730   PART 3     KA0    TESTING STARTED - RUN TIME = 02:45 MINUTES   PASSED
KA730   PART 4     KA0    TESTING STARTED - RUN TIME = 02:00 MINUTES   PASSED
KA730   PART 5     KA0    TESTING STARTED - RUN TIME = 03:00 MINUTES   PASSED
DMF32S  COMM.      XGA0   TESTING STARTED - RUN TIME = 01:00 MINUTES   PASSED
DMF32A  COMM.      TXA    TESTING STARTED - RUN TIME = 00:30 MINUTES   PASSED
DMF32P  COMM.      LCA    TESTING STARTED - RUN TIME = 00:30 MINUTES   PASSED
RL02    DISK       DLA0   TESTING STARTED - RUN TIME = 00:45 MINUTES   PASSED
R80     DISK       DQA0   TESTING STARTED - RUN TIME = 01:00 MINUTES   PASSED
RL02    DISK       DQA1   TESTING STARTED - RUN TIME = 00:45 MINUTES   PASSED

FUNCTIONAL TESTING COMPLETE - NO ERRORS
                .
                .
                .
        ( return to Main Menu )
```

3.4.2.2 TEST ALL - With Errors Printout

The following is a printout of a CRD MENU Functional Test run with failed devices detected, as a result of choosing TEST ALL from the Test Menu.

```
        ( from Test Menu )
                .
                .
                .
   Enter TEST MENU choice: D1

    Type the CTRL key and the C key (together)
    at any time to interrupt Functional Test...

    ********** BEGIN FUNCTIONAL TESTING **********

               RUN TIME = 15:00 MINUTES

KA730   PART 1     KA0    TESTING STARTED - RUN TIME = 00:45 MINUTES   PASSED
KA730   PART 2     KA0    TESTING STARTED - RUN TIME = 02:00 MINUTES   PASSED
KA730   PART 3     KA0    TESTING STARTED - RUN TIME = 02:45 MINUTES   PASSED
KA730   PART 4     KA0    TESTING STARTED - RUN TIME = 02:00 MINUTES   PASSED
KA730   PART 5     KA0    TESTING STARTED - RUN TIME = 03:00 MINUTES   PASSED
DMF32S  COMM.      XGA0   TESTING STARTED - RUN TIME = 01:00 MINUTES  *FAILED*
DMF32A  COMM.      TXA    TESTING STARTED - RUN TIME = 00:30 MINUTES   PASSED
DMF32P  COMM.      LCA    TESTING STARTED - RUN TIME = 00:30 MINUTES   PASSED
RL02    DISK       DLA0   TESTING STARTED - RUN TIME = 00:45 MINUTES  *FAILED*
R80     DISK       DQA0   TESTING STARTED - RUN TIME = 01:00 MINUTES   PASSED
RL02    DISK       DQA1   TESTING STARTED - RUN TIME = 00:45 MINUTES   PASSED

** FUNCTIONAL TESTING INCOMPLETE - ERRORS
** FAILED HARDWARE TESTING: XGA0, DLA0
** CALL FIELD SERVICE **
                .
                .
                .
        ( return to Main Menu )
```

3.4.3 CONTROL-C MENU Printout

This is the Menu printout as a result of typing the CTRL key and the C key at the same time.

(CTRL key and C key typed together)

.
.
.

```
--------------
CONTROL-C MENU
--------------

    A = Exit Menu Test

    B = Abort the current process, and return to MAIN MENU

    C = Resume the process interrupted by the CONTROL-C


    Type one of the above (e.g. A), and press RETURN.

    Enter CONTROL-C MENU choice:
```

3.4.4 FATAL ERROR Printouts

The following two printouts show what is printed when CRD MENU detects an unrecoverable error. The first printout shows what is printed when CRD MENU detects an internal software error.

.

.
:

```
THIS VERSION OF CRD MENU TEST IS CORRUPTED


MENU TEST ABORTED - CALL FIELD SERVICE
**** END OF VAX-11/730,725 MENU TEST ****

RETURNING TO VAX-11/730,725 CONSOLE, WAIT FOR  >>>' PROMPT

?02 PC=00000000
>>>
```

This printout shows what is printed when CRD MENU detects an error while the system is being sized (i.e., while the Autosizer is determining what devices are on the system).

```
                CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE

                          VERSION 1.5


**** BEGIN VAX-11/730,725 MENU TEST ****

Type the CTRL key and the C key (together) at
any time to interrupt Menu Test Processing


VAX-11/730,725 HARDWARE IDENTIFICATION  - RUN TIME = 00:30 Minutes
PLEASE WAIT . . .

UNABLE TO DETERMINE DEVICES ON THE SYSTEM - NO FURTHER TESTING POSSIBLE


MENU TEST ABORTED - CALL FIELD SERVICE
**** END OF VAX 11/730,725 MENU TEST ****

RETURNING TO VAX-11/730,725 CONSOLE, WAIT FOR  >>>  PROMPT

?02 PC=00000000
>>>
```

4 CRD ERROR DIAGNOSIS

As CRD progresses, informational messages are printed. In addition, when errors occur, CRD will always attempt to give more information concerning the condition causing the error by typing a qualifying message. However, since CRD is intended for an inexperienced operator, the level of explanation and course of action content of the message is limited. Nevertheless, there are various methods that are available to the more experienced operator to gain more information.

4.1 CRD Error Message Printout

This is applicable to both CRD AUTO and CRD MENU for informational and error messages. The operator may be able to gain more information about a condition specified by a CRD message by referring to the section in this document titled MESSAGE INDEX .

4.2 CRD Tracing Mode

This is applicable to CRD MENU only, and in particular, for errors that occur during testing of hardware devices selected for test. Since testing may fail with little or no indication as to the cause, there is a Tracing mode build into CRD MENU that can be invoked by the operator to gain more information about the error.

This Tracing mode is invoked from VAX Diagnostic Supervisor command mode (DS> prompt). When CRD is run in the Tracing mode, specific diagnostic information is printed along with the normal CRD printouts. Diagnostic information specified includes:

- VAX Diagnostic Supervisor commands used for executing the hardware unit's Diagnostic
- The actual Diagnostic error message text.

To execute CRD in this mode, type the following to the VAX Diagnostic Supervisor command level prompt (type what is underlined):

```
DS> SET CRD/TRACE
DS> CRD
```

This mode is automatically disabled when CRD is exited, or explicitly disabled by typing the following to the VAX Diagnostic Supervisor Command level prompt:

```
DS> CLEAR CRD/TRACE
```

The VAX Diagnostic Supervisor Command level can be reached by booting to the VAX Diagnostic Supervisor from the Console level (>>> prompt at terminal). See the procedure called CRD Diagnostic Supervisor Command Invocation under the section in this document titled Initiation Procedure for CRD Off-line MENU.

5 REVISION HISTORY

5.1 September 1982 Release

Submit for Release July 1982.
Initial release of CRD AUTO Offline:
Version 1.0 CRD AUTO Offline Microtest.
Version 1.0 CRD AUTO Offline Macrotest.

5.2 January 1983 Release

Submit for Release November 1982.
CRD MENU Offline Support:
Version 1.1 CRD AUTO Offline Microtest.
Version 1.1 CRD AUTO/MENU Offline Macrotest.

5.3 September 1983 Release

Submit for Release May 1983.
CRD AUTO/MENU support of UDA/RA Disk-based systems. CRD MENU Offline support for UDA/RA80,81, and 60, DZ32, DUP11, and TU80.
Version 1.2 CRD AUTO Offline Microtest.
Version 1.2 CRD AUTO/MENU Offline Macrotest.

5.4 November 1983 Release

Submit for Release July 1983.
CRD AUTO/MENU support of LESI/AZTEC Disk-based systems. AUTO/MENU support of VT100 console with summaries and pauses.
Version 1.3 CRD AUTO Offline Microtest.
Version 1.3 CRD AUTO/MENU Offline Macrotest.

5.5 March 1984 Release

Submit for Release November 1983.
CRD MENU support of TSU05. EVRMB LESI/AZTEC diagnostic does read/write testing. Documentation: SV-xxxx-xx package ordernumbers were deleted.
Version 1.4 CRD AUTO Offline Microtest.
Version 1.4 CRD AUTO/MENU Offline Macrotest.

5.6 June 1984 Release

Submit for Release October 1984.
Added VS100 and DEUNA support, removed VS125 support

6 MESSAGE INDEX

Listed below are the possible messages that may be printed during a CRD test run. Each message is listed alphabetically by the first letter in the message text. Refer to Sections titled SAMPLE PRINTOUTS for either CRD AUTO or CRD MENU, for examples of these messages in the overall context of CRD.

MESSAGES	CAUSE	SOLUTION/COMMENTS
AUTO TEST ABORTED	This message will appear as a trailing message when CRD finds an error that it cannot continue from. It may be either a Micro Diagnostic error or a Diagnostic Supervisor error.	Locate the error message that caused this message to be printed. Follow the appropriate action as described in this section.
AUTO TEST COMPLETE - ERRORS	This message will appear as a trailing message when CRD has finished testing. It indicates that all diagnostic tests for all devices selected for test have been executed, and that one or more devices have encountered errors.	Determine cause of error by locating error text in the test run printouts (procedures for further diagnosis may be found in the section called CRD Error Diagnosis found in this document). If the cause can be readily corrected, then restart CRD.
AUTO TEST COMPLETE - NO ERRORS	This message will appear as a trailing message when CRD has finished testing. It indicates that all Diagnostic tests for all devices selected for test have been executed, and that all tests have executed successfully with no errors.	Successfully completed.

MESSAGES	CAUSE	SOLUTION/COMMENTS
AUTO TEST INCOMPLETE -	This message will appear as a trailing message when CRD has finished testing. It indicates that one or more diagnostic tests for a selected device have not been executed.	Determine cause of message by locating text in the test run printouts (procedures for further diagnosis may be found in the section called CRD Error Diagnosis found in this document). If the cause can be readily corrected, then restart CRD.
AUTO TEST INCOMPLETE - ERRORS	This message will appear as a trailing message when CRD has finished testing. It indicates that one or more diagnotic tests have encountered errors.	Determine cause of message by locating text in the test run printouts (procedures for further diagnosis may be found in the section called CRD Error Diagnosis found in this document). If the cause can be readily corrected, then restart CRD.
BAD	This message is used in the event of a diagnostic error to call out the failing device.	The device(s) that are named here have failed diagnostic testing. The bad modules may be swapped out of the machine, or further testing of the device may be performed in a non-CRD environment.
%BOOT-F-Message Text	This is the format of an error encountered by VMB. They indicate that an error occured while attempting to load DIAGBOOT.EXE, the VAX Diagnostic Supervisor boot file.	The text of this message should give an indication of what the problem is. Likely causes of this error may be that DIAGBOOT was not copied contiguously onto the disk pack, or that the UNIBUS is not continuous or properly terminated. Run the EVKAA.EXE diagnostic if the problem cannot be identified from the message text.

MESSAGES	CAUSE	SOLUTION/COMMENTS
CANNOT CONTINUE CRD - MEMORY MANAGEMENT MUST BE OFF TO TURN OFF MEMORY MANAGEMENT, TYPE SET MM OFF RESTART CRD BY TYPING CRD		
	Memory Management has been turned on by the user via the SET MM ON Diagnostic Supervisor command. CRD cannot execute if Memory Management is turned on.	Turn Memory Management off and restart CRD by issuing the following commands to the Diagnostic Supervisor prompt (DS>): DS> SET MM OFF DS> CRD
CANNOT CONTINUE TESTING THIS DEVICE		
	An error occured while setting the Supervisor control or event flags, while selecting the device for testing, or while starting the test.	Run the corresponding diagnostic under the normal Diagnostic Supervisor to see the error message.
CHECK DEVICE PREPARATION - IF OK, THEN CALL FIELD SERVICE		
	This message will appear as a trailing message when CRD AUTO Test is complete. It is intended to highlight to the user that a preparation error has occured.	If the user finds that they did, in fact, set up the system correctly, they are told to call Field Service. Either the setup is still wrong, or the device to which this message refers is bad.
CPU BAD		
	Error found in Macro-Hardcore Instruction Test prior to booting the Diagnostic Supervisor. Any Macro Instruction fcund to be bad or if an unexpected machine check occured within macro-instruct'on testing.	Run EVAAA, the level 4 stand alone Hardcore Instruction Test to find what instruction failed. Possible problems are corrupted TU-58 CONSOLE Tape, or bad CPU module.

MESSAGES	CAUSE	SOLUTION/COMMENTS
CRD AUTO TEST ONLY SUPPORTED ON A VAX-11/730,725		
	CRD AUTO Test can only execute on a VAX-11/730,725. It is not currently supported for any other processor.	Make sure you are attempting to execute CRD on a VAX-11/730,725.
CRD MENU TEST ONLY SUPPORTED ON A VAX-11/730,725		
	CRD MENU Test can only execute on a VAX-11/730,725. It is not currently supported for any other processor.	Make sure you are attempting to execute CRD on a VAX-11/730,725.
?40 CRD TU-58 NOT FOUND		
	Program control was never passed to the CRD Microdiagnostic Monitor from the Console program.	Be sure the proper Micro Diagnostic tape is placed in the TU58 drive. Be sure the TU58 tape has not been corrupted. The tape drive itself or the interface logic may be broken.
DEVICE DIAGNOSTIC NOT FOUND		
	The diagnostic for the given device could not be found on the RL02 disk pack.	Refer to section titled 'Diagnostic Testing' for CRD AUTO, or 'Support' for CRD MENU, and add the diagnostic to the CRD pack.

MESSAGES	CAUSE	SOLUTION/COMMENTS
DEVICE NOT AVAILABLE		
	The Autosizer did not find the device corresponding to this message. The given port on the DMF32 (COMBO) board has been disabled via the switchpack on the distribution panel.	Be sure the device is inserted in the system. Be sure the device is receiving power. Be sure all cables, etc. to the device are properly installed. Be sure any switchpacks on the device are set for the proper address and vector. If a DMF32 port has intentionally been disabled, this is the proper message to appear. Otherwise, re-enable the port via the switchpack.
DRIVE BAD		
	This is a generic UDA50 error report given in ENKAZ to indicate the device can be identified, but the drive could not be placed on-line successfully. The identification was accomplished previously by the 'RA60, RA80, RA81 PART 1 DJAx, DUA0 TESTING STARTED' printout.	This possibly could be a controller, cabling or the specific drive which was identified previously in the 'TESTING STARTED' printout.

MESSAGES	CAUSE	SOLUTION/COMMENTS
** DRIVE NOT ONLINE - CHECK DRIVE SPUN UP ** ** DQAx: DISK DRIVE TESTING INCOMPLETE **		
	EVRAD finds an IDC disk drive not ready for testing. This should only occur for system disk testing. This message will only be displayed for Drive DQA1: if the drive is spun down following ENKCG testing.	Be sure the disk drive is up and spinning. Be sure the drive is powered on. Be sure the drive is properly connected to the IDC. Be sure all retaining screws have been removed from the disk and heads. The disk drive itself may be broken.
ENTER 1 TO RETURN TO CONSOLE, 2 TO RESTART AUTO TEST		
	CRD has found an error or a CONTROL C has been typed while the Micro Diagnostics are being run. CRD cannot continue.	Type 1 to exit CRD and return to the >>> prompt. Type 2 to restart testing at the beginning. Type 3 to see the standard Micro Diagnostic error report. If a CONTROL-C has been typed, the information displayed will indicate the test being executed when the CONTROL-C was typed.
?? ERROR IN STARTING CRD - INCORRECT BITS SET		
	An error occurred while attempting to start either CRD MENU Test or CRD AUTO Test.	This message indicates that the flag bits passed to the Diagnostic Supervisor in general register 5 were set incorrectly for CRD. The bits were set for both CRD AUTO Test and CRD MENU Test. This should only happen if the user is setting these bits manually. In this case, make sure the bits are set correctly, and restart CRD. If this happens in any of the recommended ways of starting CRD, contact Digital Field Service.

MESSAGES	CAUSE	SOLUTION/COMMENTS
?? Error in starting the VAX-11/730,725 CRD MENU TEST!		
	An error occurred in the initialization of CRD MENU Test.	Use the SET CRD/TRACE command and restart CRD. This may help determine the cause of the error. If not, call Digital Field Service.
ERROR WHILE ATTEMPTING TO EXIT CRD AUTO TEST		
	The software detected an error while trying to exit CRD.	This should never happen. Notify Digital Field Service if this error appears.
ERROR WHILE ATTEMPTING TO EXIT CRD MENU TEST		
	The software detected an error while trying to exit CRD.	This should never happen. Notify Digital Field Service if this error appears.
ERROR WHILE ATTEMPTING TO LOAD THE LOADABLE PORTION OF CRD AUTO TEST		
	A problem occured in either finding the ENSAB.EXE file on the load device, or allocating storage in memory for the control program, or in loading the file into memory.	Check to make sure the file is accessible and is not corrupted. Run the ENKCC Micro to verify that all of memory can be seen.
ERROR WHILE ATTEMPTING TO LOAD THE LOADABLE PORTION OF CRD MENU TEST		
	A problem occured in either finding the ENSAB.EXE file on the load device, or allocating storage in memory for the control program, or in loading the file into memory.	Check to make sure the file is accessible and is not corrupted. Run the ENKCC Micro to verify that all of memory can be seen.

MESSAGES	CAUSE	SOLUTION/COMMENTS
FAILED	Diagnostic testing for the device specified was not completed successfully.	Determine the cause by noting any qualifying messages in the CRD printout (procedures for further diagnosis may be found in the section called CRD Error Diagnosis found in this document). If the cause can be readily corrected, then restart CRD.
** FAILED TO ALLOCATE MEMORY	The ENSAB.EXE CRD control program encountered an error while attempting to allocate memory for its data.	Run the ENKCC Micro to verify that all of memory can be seen. If this fails to show any errors, contact Digital Field Service.
FPA NOT AVAILABLE	The FPA Micro Diagnostic (ENKCE) finds that no FPA is present in the system.	The FPA is an optional device. This is the correct message for a system with no FPA. If there is an FPA in the system, then it is broken to the point that it is not seen by the machine.
** IF DRIVE 1 IS A TS05 THEN IGNORE ERROR MESSAGE ** ** AND TYPE 2<CR> TO CONTINUE IF DRIVE 1 IS A ** ** RL02 TYPE 1 <CR> TO CONTINUE TESTING **	CRD Auto cannot distinguish between a TSU05 system and a RB730 RL02/R80 system whose RL02 is incapable of responding to even an identity check.	Type 1<CR> if the device is a TS05 and type 2<CR> if it is a RL02.

MESSAGES	CAUSE	SOLUTION/COMMENTS
INCORRECT VERSION OF CRD AUTO TEST		
	This version of CRD AUTO Test loadable image is not compatible with the version of the Diagnostic Supervisor being used.	Receive the latest released VAX Diagnostic Supervisor and CRD files (ENSAB).
INCORRECT VERSION OF CRD MENU TEST		
	This version of CRD MENU Test loadable image is not compatible with the version of the Diagnostic Supervisor being used.	Receive the latest released VAX Diagnostic Supervisor and CRD files (ENSAB).
NO SUPPORTED CONTROLLER FOUND		
	This failure indicates ENKAZ could not find an IDC or UDA50 at its base address, or there was an error in the startup initialization routine of the UDA50 controller not allowing identification of the controller.	Check that the controller is configured correctly, i.e. at the correct base CSR. Other possible problems are the controller is bad, the Unibus, or other configuration problems.
NO SUPPORTED DISK CONTROLLER FOUND BY CRD MACRO TEST		
	The VAX Diagnostic Supervisor is running in CRD AUTO mode and has completed executing EVSBA Autosizer. CRD has checked the results of EVSBA and has found that no CRD supported disk controller was identified (see section called INTRODUCTION - Determoning Base System Types and CRD AUTO OFF LINE - Support this document for supported disk controllers).	Check that hardware is configured correctly conforming to DEC Unibus I/O Address space conventions.

MESSAGES	CAUSE	SOLUTION/COMMENTS
PASSED		
	All diagnostic tests for the given device have executed successfully.	CRD begins testing the next device.
** PROPER SETUP REQUIRED - CANNOT CONTINUE ** ** CHECK DIAGNOSTIC TAPE INSERTED IN TU58 DRIVE 0 ** ** IF SETUP CORRECT, POSSIBLE TAPE OR DRIVE PROBLEM **		
	CRD MICMON has encountered a TU58 error.	Check that the TU58 tape contains all the required Micro Diagnostics. The TU58 tape is bad.
** PROPER SETUP REQUIRED - TYPE <CR> TO CONTINUE ** ** CHECK FOR RL02 DIAGNOSTIC PACK 'CRDPACK' ** ** INSERTED AND SPINNING IN DRIVE DQA1: ** ** IF SETUP CORRECT, POSSIBLE RL02 DRIVE PROBLEM **		
	The ENKCG Microdiagnostic finds Drive DQA1: not ready or contains wrong disk pack.	Be sure drive contains the RL02 disk pack labelled 'CRDPACK . Be sure the drive is up and spinning. Be sure the drive is powered on. Be sure the drive is properly connected to the IDC. Be sure all retaining screws have been removed the the drive and heads. The disk drive itself may be broken.

MESSAGES	CAUSE	SOLUTION/COMMENTS
** PROPER SETUP REQUIRED ** ** CHECK THAT DUA0: IS SPINNING ** ** THEN TYPE <CR> TO CONTINUE **	The ENKAZ level 4 diagnostic finds Drive DUA0: either RA80 or RA81 not spinning.	Be sure the drive is spun-up if this cannot be done there may be a drive problem.
** PROPER SETUP REQUIRED ** ** CHECK FOR DIAGNOSTIC PACK 'CRDPACK' ** ** INSERTED AND SPINNING IN DJAx ** ** THEN TYPE <CR> TO CONTINUE **	The ENKAZ level 4 diagnostic finds either Drive DJA0: or DJA1: RA60 not spinning or an incorrect pack via pack label.	Be sure the drive is spun-up and is the correct pack if the disk cannot be spun-up there may be a drive problem. If there is a question about the pack label if possible attempt to mount the pack under VMS. If the set-up is correct there may be a possible drive or controller problem.
** PROPER SETUP REQUIRED ** ** CHECK FOR DIAGNOSTIC PACK 'CRDPACK' ** ** INSERTED AND SPINNING IN DJAx ** ** IF SET-UP IS CORRECT, SUSPECT DRIVE ** ** PROBLEM - CALL FIELD SERVICE ** ** THEN TYPE <CR> TO CONTINUE **	The ENKAZ level 4 diagnostic finds either Drive DJA0: or DJA1: RA60 an incorrect pack via pack label and an invalid home block on the diagnostic pack.	Be sure the drive contains the correct pack. If there is a question about the pack label if possible attempt to mount the pack under VMS. If the set-up is correct there may be a possible drive or controller problem.

MESSAGES	CAUSE	SOLUTION/COMMENTS
THE ENLAA.SUP FILE COULD NOT BE FOUND! CRD MENU TEST CANNOT CONTINUE PROCESSING		
	The ENLAA.SUP support file could not be found on the load device.	Place a new, good version of ENLAA.SUP on the disk pack. Execute CRD MENU Test again. If the same message appears, contact Digital Field Service.
THE ENLAA.SUP FILE COULD NOT BE LOADED! CRD MENU TEST CANNOT CONTINUE PROCESSING		
	An error occured while loading the ENLAA.SUP support file from the load device.	Place a new, good version of ENLAA.SUP on the disk pack. Execute CRD MENU Test again. If the same message appears, contact Digital Field Service.
THE LOADABLE PORTION OF CRD AUTO TEST IS CORRUPTED		
	Either the ENSAB.EXE image is corrupted, or an internal software error was detected.	Place a new, good version of ENSAB.EXE on the disk pack. Execute CRD AUTO Test again. If the same message appears, contact Digital Field Service.
THE LOADABLE PORTION OF CRD MENU TEST IS CORRUPTED		
	Either the ENSAB.EXE image is corrupted, or an internal software error was detected.	Place a new, good version of ENSAB.EXE on the disk pack. Execute CRD MENU Test again. If the same message appears, contact Digital Field Service.
RB730 NOT AVAILABLE		
	The IDC Micro Diagnostic (ENKCF) finds that no IDC (RB730) is present in the system.	The IDC is part of the packaged system being tested by CRD. It should be present unless it is broken to the point that it cannot be recognized. Customers who buy just just the VAX-11/730,725 box product may not have an IDC present, and this is the normal message.

ZZ-
DIS

Sym

CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD
CRD

MESSAGES	CAUSE	SOLUTION/COMMENTS
The CRD Trace flag is off.		
	This is the output from the SHOW CRD Supervisor command. This is the default for the CRD Tracing mode.	No action is required.
The CRD Trace flag is on.		
	This is the output from the SHOW CRD Supervisor command. This means that the CRD Tracing mode has been enabled.	No action is required.
UNABLE TO DETERMINE DEVICES ON THE SYSTEM - NO FURTHER TESTING POSSIBLE		
	The Autosizer (EVSBA.EXE) encountered an error while attempting to identify the system devices.	Be sure the file EVSBA.EXE is on the diagnostic disk pack. Run the Autosizer under the Diagnostic Supervisor to see the error message.
UNABLE TO FIND THE LOADABLE PORTION OF CRD AUTO TEST		
	The file ENSAB.EXE was not found on the disk in DQA1:.	Be sure the RL02 pack in drive DQA1: contains all of the necessary files.
UNABLE TO FIND THE LOADABLE PORTION OF CRD MENU TEST		
	The file ENSAB.EXE was not found on the disk in DQA1:.	Be sure the RL02 pack in drive DQA1: contains all of the necessary files.

MESSAGES	CAUSE	SOLUTION/COMMENTS
UNABLE TO LOAD CONTROL PROGRAM FROM DQA1		
	This message is printed from DIAGBOOT.EXE, indicating that the Diagnostic Supervisor could not be loaded.	Be sure all files are present on the RL02 disk pack. Try to boot the Supervisor in normal mode to see the error message.
UNEXPECTED MACHINE CHECK WHILE VERIFYING LOAD PATH		
	This error is used in ENKAZ to indicate a machine check occured in the sizing operation of the program.	Try to run the EVAAA, then boot the Supervisor and run the drive specific diagnostic. If this doesn't fail run the CPU Cluster diagnostics EVKAB - EVKAE.
** UNIT IS WRITE-PROTECTED ** ** DQAx: DISK DRIVE TESTING INCOMPLETE **		
	The EVRAD diagnostic finds that one of the IDC disk drives is write-protected. EVRAD will skip write testing of the device.	For complete disk testing write enable the drive by pressing the write protect button on the drive to the unlit position.
UNSUPPORTED DISK CONFIGURATION FOUND - DRIVE 0 AND 1		
	ENKAZ cannot find a supported drive configuration or the drive cannot be identified due to a failure when an on-line command was issued. See section 2.1 for a description of the supported configurations.	Check the drive for power and that it is configured correctly, also check the cabling from the controller to the drive.

MESSAGES	CAUSE	SOLUTION/COMMENTS

```
***************************** WARNING ******************************
* MEDIA on the following hardware MUST NOT CONTAIN VALUABLE DATA *
* *
* Hardware Name Generic Name *
* ---- -------- ------------ *
* XXXXXXXXXXXX XXXXXXXXXXXX *
********************************************************************

Press RETURN when the media is ready to be tested:
```

This warning message appears when the user has selected for testing certain devices. The message pertains only to those devices listed (the 'XXXXXXXXXXXX' above). This message indicates that the devices mentioned must not contain valuable data; that is, any data on the device may be deleted when testing starts.

Place media that does not contain any valuable data (i.e., scratch media) on the device. Type a 'RETURN" to the prompt. NOTE: a CONTROL-C functions the same as a 'RETURN' for this prompt; that is, if the user types a CONTROL-C, testing will start.

?? You cannot run VAX-11/730,725 CRD MENU TEST in user mode

The user tried to execute CRD MENU Test while VAX/VMS was running.

CRD MENU Test is not supported for running under VAX/VMS. See the section on Initiation Procedures .

INDEX

DS> Diagnostic Supervisor Prompt, 57

-A-

AUTO Program Flow, 15 to 18
AUTO, CRD
  See CRD AUTO
Autosizer, 15 to 18, 56
  EVSBA, 13

-C-

CLEAR CRD/TRACE, 57
Console Program, 13, 25, 32
Console Prompt, 7, 19, 35, 57
CONTROL-C MENU Printout, 54
CRD AUTO, 1, 7, 11, 13, 15 to 29, 31 to 32, 34 to 35, 57, 59
CRD AUTO Method of Invocation, 11
CRD AUTO Off-line, 7
CRD Error Diagnosis, 57
CRD Error Message Printout, 57
CRD MENU, 1, 7, 35 to 37, 39, 46 to 57, 59
CRD MENU Fatal Software Error, 55
CRD MENU Methods of Invocation, 35
CRD MENU Off-line, 35
CRD Messages, 59
CRD Supervisor command, 35
CRD Tracing Mode, 57
CUSTOMER RUNNABLE DIAGNOSTICS
  See CRD and related topics

-D-

DEUNA, 36, 45
Diagnostic Supervisor, 13, 35, 37, 57
  See VAX Diagnostic Supervisor
Diagnostic Supervisor command file, 35
Diagnostic Testing, 39
Diagnostics, 15 to 18
Diagnostics for CRD AUTO Test, 15 to 18
Diagnostics for CRD MENU Test, 36
Disk spun up, 27, 29, 31
DMF32 (COMBO), 23 to 24, 32, 34
DMF32A, 36, 42
DMF32P, 36, 42
DMF32S, 36, 41
DR11W, 36, 44
DUP11, 36, 44 to 45
DZ32, 36, 44

-E-

ENKAX, 36, 39
ENKAZ, 13
ENSAB, 13
ENWCA, 36
EVDAB, 36, 44
EVDLB, 36, 41
EVDLC, 36, 42
EVDLD, 36, 42
EVDRE, 36, 44
EVDUP, 36, 44 to 45
EVDWA, 45
EVKAB, 36, 39
EVKAC, 36, 39
EVKAD, 36, 40
EVKAE, 36, 40
EVMAD, 36, 43
EVMAI, 43
EVMBE, 36, 44
EVQDL, 36
EVQDM, 36
EVQDQ, 36
EVQTS, 36
EVRAD, 36, 42
EVREA, 36, 40
EVREB, 36, 40
EVREC, 36, 41
EVRED, 36, 41
EVREE, 36, 41
EVRLA, 36, 43
EVRMB, 36, 43
EVSBA, 13
EVWAB, 45
Example Printout 1, 20 to 22
Example Printout 2, 24
Example Printout 3, 25
Example Printout 4, 26
Example Printout 5, 28, 30

Example Printout 6, 31
Example Printout 7, 33
Example Printout 8, 34
Exit Main Menu Test Printout, 48

-F-

Fatal Error Printouts, 55 to 56
Field Service, 1, 19
FPA Option, 20, 22

-H-

Hard Error, 32, 34
Hardware that is Tested, 36

-I-

IDENTIFICATION, 1 to 2
Informational Messages, 57
Initiation Procedure, 11, 37
INTRODUCTION, 1

-K-

KA730 CPU, 36, 39 to 40

-M-

MAIN MENU, 48 to 51
MAIN MENU (Functional Test)
Printout, 47
MENU (Functional Test) Printout,
MAIN, 47
MENU Printout, CONTROL-C, 54
MENU Printout, TEST, 51
MENU, CRD
See CRD MENU
Message Index, 59
Methods of Invocation, 35
MICMON
See VAX-11/730,725
Microdiagnostic Monitor
Micro Diagnostics
See VAX-11/730,725
Microdiagnostic Monitor
Microcode, 11
Microdiagnostic Monitor
See VAX-11/730,725
Microdiagnostic Monitor
Missing Device, 23

-P-

Print Hardware Preparation
Printout, 50
Print System Hardware Printout,
49
Printout, CONTROL-C MENU, 54
Printout, exit Main Menu Test, 48
Printout, MAIN MENU (Functional
Test), 47
Printout, print Hardware
Preparation, 50
Printout, print System Hardware,
49
Printout, TEST ALL - No Errors,
52
Printout, TEST ALL - With Errors,
53
Printout, TEST MENU, 51
Printouts, fatal Error, 55
Program Flow - 'T' Console
Command, 13

-R-

R80, 42
RA60, 43
RA80, 43
RA81, 43
RB730, 23, 32
RB730/RL02,R80, 36
RC25, 2, 5, 8, 11, 18, 21, 37, 43
RCF25, 2, 5, 8, 11, 18, 21, 37,
43
RK07, 42
RK711, 36, 40 to 41
RK711/RK07, 36
RL02, 25, 27, 36, 42
RL211/RL02, 36

-S-

Sample Printouts, 19, 46
SET CRD/TRACE, 57
Subsystems, 35
Support - Hardware/Diagnostic, 36
Synchronous and Parallel Ports,
23

-T-

TEST ALL - No Errors Printout, 52

TEST ALL - With Errors Printout, 53
TEST MENU, 51 to 53
TEST MENU Printout, 51
TRACE, 57
TS05, 36, 43
TS11, 36, 43
TU58, 11, 13, 25 to 26
TU80, 36, 43

U-

UDA/RA, 36
User Document, 1

-V-

VAX Diagnostic Supervisor, 13, 15 to 18, 35, 39, 57
Diagnostic Supervisor, 1
VDS, 1
VAX-11/730,725, 7, 11
VAX-11/730,725 Microdiagnostic Monitor, 13
MICMON, 1
Micro Diagnostics, 1
Microdiagnostic Monitor, 1
VDS
See VAX Diagnostic Supervisor
VS100, 36, 45

-W-

Write-protect, 11
Write-Protected Disks, 29

[End of ENSAB.DOC - CUSTOMER RUNNABLE DIAGNOSTICS]

Object Module Synopsis

Module Name	Ident	Bytes	File	Creation Date	Creator
CRDACT1	01-22	1256	[DS.CRD.V020]CRDACT1.OBJ;1	19-Sep-1985 16:31	VAX-11 Bliss-32 V4.1-746
CRDACT2	01-17	1981	[DS.CRD.V020]CRDACT2.OBJ;1	19-Sep-1985 16:32	VAX-11 Bliss-32 V4.1-746
CRDCTRLC	01-00	83	[DS.CRD.V020]CRDCTRLC.OBJ;1	19-Sep-1985 16:34	VAX-11 Bliss-32 V4.1-746
CRDDDB	01-11	6170	[DS.CRD.V020]CRDDDB.OBJ;1	19-Sep-1985 16:35	VAX-11 Bliss-32 V4.1-746
CRDGENCOM	01-02	516	[DS.CRD.V020]CRDGENCOM.OBJ;1	19-Sep-1985 16:49	VAX-11 Bliss-32 V4.1-746
CRDHDWSUP	01-05	5663	[DS.CRD.V020]CRDHDWSUP.OBJ;1	19-Sep-1985 16:50	VAX-11 Bliss-32 V4.1-746
CRDHSACT	01-01	1059	[DS.CRD.V020]CRDHSACT.OBJ;1	19-Sep-1985 16:52	VAX-11 Bliss-32 V4.1-746
CRDINIT	01-06	213	[DS.CRD.V020]CRDINIT.OBJ;1	19-Sep-1985 16:53	VAX-11 Bliss-32 V4.1-746
CRDINTRFC	07	1000	[DS.CRD.V020]CRDINTRFC.OBJ;1	19-Sep-1985 16:54	VAX-11 Bliss-32 V4.1-746
CRDTEXT	11	5581	[DS.CRD.V020]CRDTEXT.OBJ;1	19-Sep-1985 16:55	VAX-11 Bliss-32 V4.1-746
CRDMMACT	V01.4	2571	[DS.CRD.V020]CRDMMACT.OBJ;1	19-Sep-1985 16:56	VAX-11 Bliss-32 V4.1-746
CRDPTABLE	01-01	545	[DS.CRD.V020]CRDPTABLE.OBJ;1	19-Sep-1985 16:59	VAX-11 Bliss-32 V4.1-746
CRDSTATE	01-04	7860	[DS.CRD.V020]CRDSTATE.OBJ;1	19-Sep-1985 17:00	VAX-11 Bliss-32 V4.1-746
CRDSTOP	V01.20	2375	[DS.CRD.V020]CRDSTOP.OBJ;1	19-Sep-1985 17:06	VAX-11 Bliss-32 V4.1-746
CRDTQACT	01-00	479	[DS.CRD.V020]CRDTQACT.OBJ;1	19-Sep-1985 17:11	VAX-11 Bliss-32 V4.1-746
CRDTYPCOD	01-01	967	[DS.CRD.V020]CRDTYPCOD.OBJ;1	19-Sep-1985 17:11	VAX-11 Bliss-32 V4.1-746
CRDVECS	V1.09	104	[DS.CRD.V020]CRDVECS.OBJ;1	19-SEP-1985 17:14	VAX/VMS Macro V04-00
MENCNTLC	01-00	779	[DS.CRD.V020]MENCNTLC.OBJ;1	19-Sep-1985 17:14	VAX-11 Bliss-32 V4.1-746
MENGOA	NONE	2068	[DS.CRD.V020]MENGOA.OBJ;1	19-Sep-1985 17:15	VAX-11 Bliss-32 V4.1-746
MENPROMPT	V01.1	3093	[DS.CRD.V020]MENPROMPT.OBJ;1	19-Sep-1985 17:16	VAX-11 Bliss-32 V4.1-746
MENSTATE	V01.4	7949	[DS.CRD.V020]MENSTATE.OBJ;1	19-Sep-1985 17:20	VAX-11 Bliss-32 V4.1-746
MENTST	01-04	2027	[DS.CRD.V020]MENTST.OBJ;1	19-Sep-1985 17:21	VAX-11 Bliss-32 V4.1-746
MENTSTINI	V01.4	2834	[DS.CRD.V020]MENTSTINI.OBJ;1	19-Sep-1985 17:23	VAX-11 Bliss-32 V4.1-746
MENTSTQ	01 00	202	[DS.CRD.V020]MENTSTQ.OBJ;1	19-Sep-1985 17:24	VAX-11 Bliss-32 V4.1-746
MENTURING	V01.2	4192	[DS.CRD.V020]MENTURING.OBJ;1	19-Sep-1985 17:25	VAX-11 Bliss-32 V4.1-746
LIB$SPAWN	1-005	366	SYS$COMMON:[SYSLIB]STARLET.OLB;5	23-Jun-1985 01:51	VAX-11 Bliss-32 V4.0-742
LIB$MSGDEF	2-018	0	SYS$COMMON:[SYSLIB]STARLET.OLB;5	23-JUN-1985 00:54	VAX-11 Message V04-00
SYS$P1_VECTOR	V04 000	0	SYS$COMMON:[SYSLIB]STARLET.OLB;5	23-JUN-1985 01:59	VAX/VMS Macro V04-00
LIB$ANALYZE_SDESC	1 003	101	SYS$COMMON:[SYSLIB]STARLET.OLB;5	23-JUN-1985 00:57	VAX/VMS Macro V04-00
LIB$MSGDEF	2 018	0	SYS$COMMON:[SYSLIB]STARLET.OLB;5	23-JUN-1985 00:54	VAX-11 Message V04-00

Image Section Synopsis

Cluster	Type	Pages	Base Addr	Disk	VBN	PFC	Protection and Paging	Global Sec. Name	Match	Major'd	Minorid
DEFAULT_CLUSTER	0	122	00000000		1	0	READ ONLY				

Key for special characters above:

R	- Relocatable
P	- Protected

Program Section Synopsis

Psect Name	Module Name	Base	End	Length	Align	Attributes
$BASE		00000000	00000067	00000068 (104.)	LONG 2	NOPIC,USR,CON,REL,LCL, SHR,NOEXE, RD, WRT,NOVEC
	CRDVECS	00000000	00000067	00000068 (104.)	LONG 2	
$CODE$		00000068	0000046B	00000404 (1028.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD,NOWRT,NOVEC
	MENGOA	00000068	0000046B	00000404 (1028.)	LONG 2	
OWN		0000046C	000004DF	00000074 (116.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR,NOEXE, RD, WRT,NOVEC
	MENGOA	0000046C	000004DF	00000074 (116.)	LONG 2	
$PLIT$		000004E0	0000087B	0000039C (924.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR,NOEXE, RD,NOWRT,NOVEC
	MENGOA	000004E0	0000087B	0000039C (924.)	LONG 2	
CODE		0000087C	00006E07	0000658C (25996.)	LONG 2	PIC,USR,CON,REL,LCL, SHR, EXE, RD,NOWRT,NOVEC
	CRDACT1	0000087C	00000BF6	0000037B (891.)	LONG 2	
	CRDACT2	00000BF8	00001290	00000699 (1689.)	LONG 2	
	CRDCTRLC	00001294	000012DD	0000004A (74.)	LONG 2	
	CRDDDB	000012E0	00002052	00000D73 (3443.)	LONG 2	
	CRDGENCOM	00002054	000020DE	0000008B (139.)	LONG 2	
	CRDHDWSUP	000020E0	00002B07	00000A28 (2600.)	LONG 2	
	CRDHSACT	00002B08	00002EA4	0000039D (925.)	LONG 2	
	CRDINIT	00002EA8	00002F6C	000000C5 (197.)	LONG 2	
	CRDINTRFC	00002F70	0000319F	00000230 (560.)	LONG 2	
	CRDMMACT	000031A0	000037E7	00000648 (1608.)	LONG 2	
	CRDPTABLE	000037E8	00003936	0000014F (335.)	LONG 2	
	CRDSTATE	00003938	00003E19	000004E2 (1250.)	LONG 2	
	CRDSTOP	00003E1C	0000462D	00000812 (2066.)	LONG 2	
	CRDTQACT	00004630	00004760	00000131 (305.)	LONG 2	
	CRDTYPCOD	00004764	00004919	000001B6 (438.)	LONG 2	
	MENCNTLC	0000491C	00004BF8	000002DD (733.)	LONG 2	
	MENPROMPT	00004BFC	000056A3	00000AA8 (2728.)	LONG 2	
	MENTST	000056A4	00005D11	0000066E (1646.)	LONG 2	
	MENTSTINI	00005D14	00006516	00000803 (2051.)	LONG 2	
	MENTSTQ	00006518	000065C1	000000AA (170.)	LONG 2	
	MENTURING	000065C4	00006E07	00000844 (2116.)	LONG 2	
DATA		00006E08	0000E65A	00007853 (30803.)	LONG 2	NOPIC,USR,CON,REL,LCL, SHR,NOEXE, RD,NOWRT,NOVEC
	CRDACT1	00006E08	00006F74	0000016D (365.)	LONG 2	
	CRDACT2	00006F78	0000709B	00000124 (292.)	LONG 2	
	CRDCTRLC	0000709C	000070A4	00000009 (9.)	LONG 2	
	CRDDDB	000070A8	00007B46	00000A9F (2719.)	LONG 2	
	CRDGENCOM	00007B48	00007CBC	00000175 (373.)	LONG 2	
	CRDHDWSUP	00007CC0	0000856E	000008AF (2223.)	LONG 2	
	CRDHSACT	00008570	000085F5	00000086 (134.)	LONG 2	
	CRDINIT	000085F8	000085FF	00000008 (8.)	LONG 2	
	CRDINTRFC	00008600	00008687	00000088 (136.)	LONG 2	
	CRDTEXT	00008688	00009BD0	00001549 (5449.)	LONG 2	

Psect Name	Module Name	Base	End	Length		Align	Attributes
DATA		00006E08	0000E65A	00007853	(30803.)	LONG 2	NOPIC,USR,CON,REL,LCL, SHR,NOEXE, RD,NOWRT,NOVEC
	CRDMMACT	00009BD4	00009DA2	000001CF	(463.)	LONG 2	
	CRDPTABLE	00009DA4	00009E69	000000C6	(198.)	LONG 2	
	CRDSTATE	00009E6C	0000B82D	000019C2	(6594.)	LONG 2	
	CRDSTOP	0000B830	0000B8FC	000000CD	(205.)	LONG 2	
	CRDTQACT	0000B900	0000B9AD	000000AE	(174.)	LONG 2	
	CRDTYPCOD	0000B9B0	0000BBC0	00000211	(529.)	LONG 2	
	MENCNTLC	0000BBC4	0000BBCE	0000000B	(11.)	LONG 2	
	MENPROMPT	0000BBD0	0000BBDB	0000000C	(12.)	LONG 2	
	MENSTATE	0000BBDC	0000DAC7	00001EEC	(7916.)	LONG 2	
	MENTST	0000DAC8	0000DB44	0000007D	(125.)	LONG 2	
	MENTSTINI	0000DB48	0000DE1E	000002D7	(727.)	LONG 2	
	MENTSTQ	0000DE20	0000DE3F	00000020	(32.)	LONG 2	
	MENTURING	0000DE40	0000E65A	0000081B	(2075.)	LONG 2	
WORK		0000E65C	0000F0C8	00000A6D	(2669.)	LONG 2	NOPIC,USR,CON,REL,LCL,NOSHR,NOEXE, RD, WRT,NOVEC
	CRDDDB	0000E65C	0000E663	00000008	(8.)	LONG 2	
	CRDGENCOM	0000E664	0000E667	00000004	(4.)	LONG 2	
	CRDHDWSUP	0000E668	0000E9AF	00000348	(840.)	LONG 2	
	CRDINIT	0000E9B0	0000E9B7	00000008	(8.)	LONG 2	
	CRDINTRFC	0000E9B8	0000EAE7	00000130	(304.)	LONG 2	
	CRDTEXT	0000EAE8	0000EB6B	00000084	(132.)	LONG 2	
	CRDMMACT	0000EB6C	0000ED5F	000001F4	(500.)	LONG 2	
	CRDPTABLE	0000ED60	0000ED6B	0000000C	(12.)	LONG 2	
	CRDSTATE	0000ED6C	0000ED7B	00000010	(16.)	LONG 2	
	CRDSTOP	0000ED7C	0000EDE3	00000068	(104.)	LONG 2	
	MENCNTLC	0000EDE4	0000EE06	00000023	(35.)	LONG 2	
	MENPROMPT	0000EE08	0000EF68	00000161	(353.)	LONG 2	
	MENSTATE	0000EF6C	0000EF8C	00000021	(33.)	LONG 2	
	MENTST	0000EF90	0000F08F	00000100	(256.)	LONG 2	
	MENTSTINI	0000F090	0000F0C7	00000038	(56.)	LONG 2	
	MENTURING	0000F0C8	0000F0C8	00000001	(1.)	LONG 2	
LIB$CODE		0000F0CC	0000F2A0	000001D5	(469.)	LONG 2	PIC,USR,CON,REL,LCL, SHR, EXE, RD,NOWRT,NOVEC
	LIB$SPAWN	0000F0CC	0000F239	0000016E	(366.)	LONG 2	
	LIB$ANALYZE_SDESC	0000F23C	0000F2A0	00000065	(101.)	LONG 2	
_LIB$DATA		0000F2A4	0000F2A4	00000000	(0.)	LONG 2	PIC,USR,CON,REL,LCL,NOSHR,NOEXE, RD, WRT,NOVEC
	LIB$ANALYZE_SDESC	0000F2A4	0000F2A4	00000000	(0.)	LONG 2	

Symbol Cross Reference

Symbol	Value	Defined By	Referenced By ...	
CRD$ADVANCE_DIAGNOSTIC	00000BF8-R	CRDACT2	CRDACT2	CRDSTATE
CRD$ADVANCE_GENERAL_COM	00000C84-R	CRDACT2	CRDACT2	CRDSTATE
CRD$AL_CRD_DISPATCH_VECTORS	00000000-R	CRDVECS	CRDACT1	CRDMMACT
			CRDSTOP	MENCNTLC
			MENPROMPT	
CRD$AL_IDC_HDWR_SPRT_TABLE	0000E670-R	CRDHDWSUP	CRDACT2	CRDDDB
			CRDHDWSUP	
CRD$AL_LESI_HDWR_SPRT_TABLE	0000E710-R	CRDHDWSUP	CRDACT2	CRDDDB
			CRDHDWSUP	
CRD$AL_UDA_0_HDWR_SPRT_TABLE	0000E770-R	CRDHDWSUP	CRDACT2	CRDDDB
			CRDHDWSUP	
CRD$AL_UDA_1_HDWR_SPRT_TABLE	0000E7F0-R	CRDHDWSUP	CRDACT2	CRDDDB
			CRDHDWSUP	
CRD$AL_UDA_2_HDWR_SPRT_TABLE	0000E890-R	CRDHDWSUP	CRDACT2	CRDDDB
			CRDHDWSUP	
CRD$AL_UDA_3_HDWR_SPRT_TABLE	0000E910-R	CRDHDWSUP	CRDACT2	CRDDDB
			CRDHDWSUP	
CRD$ASSIGN_PTABLE_PTRS	00000CC0-R	CRDACT2	CRDACT2	CRDSTATE
CRD$AUTOSIZER_ERROR	0000087C-R	CRDACT1	CRDACT1	CRDMMACT
			CRDSTATE	
CRD$AUTOSIZER_SET_FLAGS	00000D10-R	CRDACT2	CRDACT2	CRDSTATE
CRD$AUTO_SUMMARY	0000418A-R	CRDSTOP	CRDSTOP	
CRD$BUILD_DDBS	000012E0-R	CRDDDB	CRDACT2	CRDDDB
CRD$BUILD_SUPPORT_TABLE	000020E0-R	CRDHDWSUP	CRDHDWSUP	CRDINIT
CRD$CHECK_CTRL_C	000012D4-R	CRDCTRLC	CRDCTRLC	
CRD$CLEAR_CTRL_C_FLAG	000012CC-R	CRDCTRLC	CRDCTRLC	CRDSTOP
			MENCNTLC	
CRD$CLEAR_CTRL_C_HANDLERS	000012A3-R	CRDCTRLC	CRDCTRLC	MENCNTLC
CRD$COMMON_INITIALIZATIONS	00002F03-R	CRDINIT	CRDINIT	MENTSTINI
CRD$CONTROL_C_HANDLER	00004174-R	CRDSTOP	CRDINIT	CRDSTOP
CRD$CRD_INITIALIZATION	00002EA8-R	CRDINIT	CRDINIT	CRDINTRFC
CRD$DETERMINE_BASE_SYSTEM	00000D5D-R	CRDACT2		
CRD$DIAGNOSTIC_ERROR	0000089C-R	CRDACT1	CRDACT1	CRDSTATE
CRD$DISABLE_CTRL_C	000012BA-R	CRDCTRLC	CRDCTRLC	CRDSTOP
			MENCNTLC	MENPROMPT
CRD$DISPOSE	00001F9F-R	CRDDDB	CRDDDB	CRDSTOP
			MENTSTINI	
CRD$DONE GENERAL_COMS	00000E9F-R	CRDACT2	CRDACT2	CRDHSACT
			CRDSTATE	
CRDDSGA_PTDESC	0000003C-R	CRDVECS		
CRDDSRCHECK_AUTOTEST_OFF_SET	00000038-R	CRDVECS	CRDSTOP	
CRDDSRCHECK_MENUTEST_OFF_SET	00000034-R	CRDVECS	CRDSTOP	
CRD$DS_INTERFACE	00002F70-R	CRDINTRFC	CRDINTRFC	CRDSTOP
			CRDVECS	
CRD$ENABLE CTRL C	000012B0-R	CRDCTRLC	CRDCTRLC	CRDINIT
			MENCNTLC	MENPROMPT
CRDEXEALONONPAGED	00000018-R	CRDVECS	CRDDDB	MENTSTINI

Symbol	Value	Defined By	Referenced By ...	
CRDEXEDEANONPAGED	00000024-R	CRDVECS	CRDDDB	
CRD$EXIT_CRD	000008F6-R	CRDACT1	CRDACT1	CRDSTATE
CRD$EXIT_DS	00000004-R	CRDVECS	CRDSTOP	
CRD$FIND_PTABLE	000037E8-R	CRDPTABLE	CRDACT2	CRDPTABLE
CRD$GAW_HS_STATE_TABLE	0000CDB8-R	MENSTATE	MENSTATE	MENTURING
CRD$GAW_MM_STATE_TABLE	0000BBE8-R	MENSTATE	MENSTATE	MENTURING
CRD$GAW_TQ_STATE_TABLE	0000C7C8-R	MENSTATE	MENSTATE	MENTURING
CRD$GA_BEGIN	0000002C-R	CRDVECS	CRDSTOP	MENCNTLC
CRD$GA_CRD_MESSAGE_BUFFER	0000EAE8-R	CRDTEXT	CRDACT2	CRDSTOP
			CRDTEXT	
CRD$GA_CURRENT_DIAGNOSTIC	0000E65C-R	CRDDDB	CRDACT1	CRDACT2
			CRDDDB	CRDHSACT
CRD$GA_DS_CTRL_C_FIRST	00000010-R	CRDVECS	CRDCTRLC	
CRD$GA_DS_CTRL_C_SECOND	00000028-R	CRDVECS	CRDCTRLC	
CRD$GA_DS_FLAGS	0000000C-R	CRDVECS	CRDCTRLC	CRDMMACT
			MENTURING	
CRD$GA_FILE_FAB	0000EBF8-R	CRDMMACT	CRDMMACT	
CRD$GA_FILE_RAB	0000EC48-R	CRDMMACT	CRDMMACT	
CRD$GA_FIRST_DIAGNOSTIC	0000E660-R	CRDDDB	CRDACT1	CRDACT2
			CRDDDB	CRDSTOP
CRD$GA_GENERAL_COMMANDS	00007B54-R	CRDGENCOM	CRDGENCOM	
CRD$GA_HDWR_SPRT_TABLE	0000E66C-R	CRDHDWSUP	CRDACT2	CRDHDWSUP
CRD$GA_HS_CURRENT_DDB_PTR	0000EF80-R	MENSTATE	CRDHSACT	CRDMMACT
			CRDTQACT	MENSTATE
CRD$GA_HS_CURRENT_HSB_PTR	0000EF84-R	MENSTATE	CRDHSACT	CRDMMACT
			MENSTATE	
CRD$GA_PTABLE	00000014-R	CRDVECS	CRDPTABLE	MENTSTINI
CRD$GA_REC_BUF	0000EB84-R	CRDMMACT	CRDMMACT	MENGOA
CRD$GA_USER_ERROR_TEXT	0000001C-R	CRDVECS	CRDACT1	CRDHSACT
CRD$GB_AUTO_TO_MENU_OFF	0000ED7C-R	CRDSTOP		
CRD$GB_BASE_SYSTEM	0000E668-R	CRDHDWSUP	CRDACT2	CRDDDB
			CRDHDWSUP	
CRD$GB_CRD_DEBUG	00000030-R	CRDVECS	CRDACT1	CRDACT2
			CRDDDB	CRDHSACT
			CRDINIT	CRDSTATE
			CRDTQACT	MENPROMPT
			MENTST	MENTSTINI
			MENTURING	
CRD$GB_CURRENT_STATE_TABLE	0000EF8C-R	MENSTATE	CRDHSACT	CRDMMACT
			CRDTQACT	MENCNTLC
			MENSTATE	MENTSTINI
			MENTURING	
CRD$GB_USER_PROMPT_OKAY	0000EA4C-R	CRDINTRFC	CRDINTRFC	CRDSTOP
			MENCNTLC	
CRD$GET_ACTION_ROUTINE	00003988-R	CRDSTATE	CRDSTATE	
CRD$GET_DEVICE_NAME	00003835-R	CRDPTABLE	CRDHSACT	CRDPTABLE
			CRDSTOP	MENPROMPT
			MENTST	
CRD$GET_GENERAL_COMMAND	0000205D-R	CRDGENCOM	CRDACT2	CRDGENCOM
CRD$GET_STATE	0000394F-R	CRDSTATE	CRDSTATE	

Symbol	Value	Defined By	Referenced By ...	
CRD$GL_CRD_DEBUG_VECTOR	0000ED80-R	CRDSTOP	CRDACT1	CRDMMACT
			CRDSTOP	
CRD$GL_CURRENT_ACTION	0000ED74-R	CRDSTATE	CRDACT1	CRDSTATE
			MENTURING	
CRD$GL_CURRENT_GENERAL_COM	0000E664-R	CRDGENCOM	CRDACT2	CRDGENCOM
			CRDHSACT	
CRD$GL_CURRENT_STATE	0000ED70-R	CRDSTATE	CRDACT1	CRDACT2
			CRDSTATE	
CRD$GL_CURRENT_TYPECODE	0000ED6C-R	CRDSTATE	CRDACT1	CRDSTATE
			MENTURING	
CRD$GL_HS_CURRENT_HSB_NUMB	0000EF88-R	MENSTATE	CRDHSACT	CRDMMACT
			MENSTATE	
CRD$GL_HS_CURRENT_STATE	0000EF78-R	MENSTATE	CRDHSACT	CRDMMACT
			MENCNTLC	MENSTATE
			MENTURING	
CRD$GL_MM_CURRENT_STATE	0000EF70-R	MENSTATE	CRDMMACT	MENCNTLC
			MENSTATE	MENTURING
CRD$GL_PREVIOUS_STATE	0000EF6C-R	MENSTATE	CRDMMACT	MENSTATE
			MENTURING	
CRD$GL_SAVED_DSA_FLAGS	0000E9B4-R	CRDINIT	CRDINIT	
CRD$GL_TQ_CURRENT_STATE	0000EF74-R	MENSTATE	CRDMMACT	CRDTQACT
			MENCNTLC	MENSTATE
			MENTURING	
CRD$GL_TQ_CURRENT_TQ_ENTRY	0000EF7C-R	MENSTATE	CRDMMACT	CRDTQACT
			MENSTATE	
CRD$GT_2_TABS	000086A3-R	CRDTEXT	CRDTEXT	
CRD$GT_ALL_FAILURES	00008B35-R	CRDTEXT	CRDSTOP	CRDTEXT
			MENTST	
CRD$GT_ALL_INC_HDWR_PREP	00008B91-R	CRDTEXT	CRDSTOP	CRDTEXT
			MENTST	
CRD$GT_ALL_OTHERS	00008BD7-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_ALL_PASSES	00008B63-R	CRDTEXT	CRDSTOP	CRDTEXT
			MENTST	
CRD$GT_AUTO	000086AD-R	CRDTEXT	CRDINIT	CRDTEXT
CRD$GT_AUTOSIZER_ERROR	000089F7-R	CRDTEXT	CRDACT1	CRDMMACT
			CRDTEXT	
CRD$GT_AUTOSIZER_NAME	00009BDD-R	CRDMMACT	CRDACT2	CRDMMACT
CRD$GT_AUTOSIZER_SET_FLAGS	00009BE3-R	CRDMMACT	CRDACT2	CRDMMACT
CRD$GT_CANNOT_LOAD_DIAG	00008A73-R	CRDTEXT	CRDACT1	CRDTEXT
CRD$GT_CONTROL_C_ABORT	000088B8-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_CRD_BAD_DEVICE	00008743-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_CRD_DEV_UNTESTED	000087BD-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_CRD_END_MESSAGE	000088EB-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_CRD_ERRORS_COMPLETE	00008724-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_CRD_EVSBA_ERROR	00008810-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_CRD_EVSBB_ERROR	0000883A-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_CRD_FATAL_ERROR	00008864-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_CRD_INCOMPLETE	0000877A-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_CRD_NO_ERRORS	00008702-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_DEV_UNAVAILABLE	00008A43-R	CRDTEXT	CRDACT2	CRDTEXT

Symbol	Value	Defined By	Referenced By ...	
CRD$GT_DEV_UNAVAILABLE_2	00008A5C-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_DS_COMMAND_OUT	00008AF5-R	CRDTEXT	CRDHSACT	CRDTEXT
CRD$GT_DS_LOAD_COM	000084B6-R	CRDHDWSUP	CRDACT2	CRDHDWSUP
			CRDHSACT	CRDMMACT
CRD$GT_DS_SELECT_COM	000084D8-R	CRDHDWSUP	CRDACT2	CRDHDWSUP
			CRDHSACT	
CRD$GT_DS_SET_EVENT_COM	000084C7-R	CRDHDWSUP	CRDACT2	CRDHDWSUP
			CRDHSACT	
CRD$GT_DS_SET_FLAGS_COM	000084BC-R	CRDHDWSUP	CRDACT2	CRDHDWSUP
			CRDHSACT	CRDMMACT
CRD$GT_DS START_COM	000084E0-R	CRDHDWSUP	CRDACT2	CRDHDWSUP
			CRDHSACT	CRDMMACT
CRD$GT_EXIT_TO_CLI	0000894F-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_EXIT TO CONSOLE	00008912-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_FAIL	000086E6-R	CRDTEXT	CRDACT1	CRDHSACT
			CRDTEXT	
CRD$GT_INIT_ALL_PASSES	00008BFD-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_INTERNAL_ERROR	00008AC5-R	CRDTEXT	CRDSTOP	CRDTEXT
			MENTST	
CRD$GT_INTERNAL_ERROR_ABORT	0000888E-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_INV_BASE_SYSTEM	000089C0-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_MENU	000086A8-R	CRDTEXT	CRDINIT	CRDTEXT
CRD$GT_NEW_LINE	00008696-R	CRDTEXT	CRDACT1	CRDSTOP
			CRDTEXT	MENPROMPT
			MENTST	
CRD$GT_NEW LINE_MESSAGE	00008B14-R	CRDTEXT	CRDSTOP	CRDTEXT
			MENCNTLC	MENPROMPT
			MENTST	MENTSTINI
CRD$GT_PASS	000086F4-R	CRDTEXT	CRDACT2	CRDHSACT
			CRDTEXT	
CRD$GT_PREP_ERROR_SPACES	00008ABB-R	CRDTEXT	CRDACT1	CRDTEXT
CRD$GT_PRESS_RETURN	00008989-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_PRESS_RETURN_MM	000089A2-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_RUNTIME	000086D8-R	CRDTEXT	CRDACT2	CRDTEXT
CRD$GT_SAME_LINE_MESSAGE	00008B13-R	CRDTEXT	CRDTEXT	
CRD$GT_SLASH	0000869C-R	CRDTEXT	CRDSTOP	CRDTEXT
			MENTST	
CRD$GT_SPACE_AND_SPACE	00008690-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT_START_ERROR	00008A93-R	CRDTEXT	CRDACT1	CRDTEXT
CRD$GT_STAR_LINE_PREFIX_MESSAGE	00008B17-R	CRDTEXT	CRDTEXT	
CRD$GT_STAR_LINE_SUFFIX_MESSAGE	00008B1D-R	CRDTEXT	CRDTEXT	
CRD$GT_SUMMARY_TITLE	00008B21-R	CRDTEXT	CRDSTOP	CRDTEXT
			MENTST	
CRD$GT_TAB	00008699-R	CRDTEXT	CRDSTOP	CRDTEXT
			MENTST	
CRD$GT_TESTING_DEVICE	000086B2-R	CRDTEXT	CRDACT2	CRDTEXT
CRD$GT_TESTING_STARTED	000086BA-R	CRDTEXT	CRDACT2	CRDTEXT
CRD$GT UNKNOWN	0000869E-R	CRDTEXT	CRDSTOP	CRDTEXT
CRD$GT WHICH CRD	0000E9B0-R	CRDINIT	CRDINIT	CRDSTOP
CRD$GW STATE TABLE	00009E6C-R	CRDSTATE	CRDSTATE	

Symbol	Value	Defined By	Referenced By ...	
CRD$INIT_DDB_STATUS	00001FD9-R	CRDDDB	CRDDDB	CRDHSACT
			CRDTQACT	
CRD$INIT_GENERAL_COM_TABLE	00002054-R	CRDGENCOM	CRDGENCOM	CRDINIT
			MENTSTINI	
CRD$INIT_STATE_TABLE	00003938-R	CRDSTATE	CRDINIT	CRDSTATE
CRD$INTERNAL_ERROR	000009BA-R	CRDACT1	CRDACT1	CRDACT2
			CRDDDB	CRDSTATE
			MENTURING	
CRD$LEVEL_4_PASSED	00000EC9-R	CRDACT2	CRDACT2	CRDSTATE
CRD$LOAD_AUTOSIZER	00000EE0-R	CRDACT2	CRDACT2	CRDSTATE
CRD$LOAD_ERROR	00000A34-R	CRDACT1	CRDACT1	CRDHSACT
			CRDSTATE	
CRD$LOAD_GENERAL_COM	00000F27-R	CRDACT2	CRDACT2	CRDHSACT
			CRDSTATE	
CRD$LOAD_LOAD_COM	00000F6C-R	CRDACT2	CRDACT2	CRDSTATE
CRD$LOAD_SELECT_COM	00000FC0-R	CRDACT2	CRDACT2	CRDSTATE
CRD$LOAD_SET_EVENT_COM	00001017-R	CRDACT2	CRDACT2	CRDSTATE
CRD$LOAD_SET_FLAGS_COM	0000106E-R	CRDACT2	CRDACT2	CRDSTATE
CRD$LOAD_START_COM	000010C5-R	CRDACT2	CRDACT2	CRDSTATE
CRD$LOG_START	000030B3-R	CRDINTRFC		
CRD$LOG_STOP	00003154-R	CRDINTRFC		
CRD$LOG_WRITE	00003172-R	CRDINTRFC		
CRD$NEW	00001FEC-R	CRDDDB	CRDDDB	
CRD$PREPARATION_ERROR	00000A68-R	CRDACT1	CRDACT1	CRDSTATE
CRD$PRINT	00000008-R	CRDVECS	CRDACT1	CRDACT2
			CRDDDB	CRDGENCOM
			CRDHDWSUP	CRDHSACT
			CRDMMACT	CRDPTABLE
			CRDSTATE	CRDSTOP
			CRDTQACT	CRDTYPCOD
			MENCNTLC	MENPROMPT
			MENTST	MENTSTINI
			MENTURING	
CRD$PRINT_ACTION_ROUTINE	00003C31-R	CRDSTATE	CRDSTATE	
CRD$PRINT_COMMAND_TABLE	00002081-R	CRDGENCOM	CRDGENCOM	CRDINIT
CRD$PRINT_DDB	00000B27-R	CRDACT1	CRDACT1	CRDACT2
			CRDSTOP	
CRD$PRINT_DS_COMMAND	00002E85-R	CRDHSACT	CRDACT2	CRDHSACT
			CRDMMACT	
CRD$PRINT_HARDWARE_SUPPORT	00002A95-R	CRDHDWSUP	CRDHDWSUP	CRDINIT
CRD$PRINT_PTABLES	00003865-R	CRDPTABLE	CRDINIT	CRDPTABLE
CRD$PRINT_TYPECODE_NAME	00004764-R	CRDTYPCOD	CRDSTATE	CRDTYPCOD
			MENTURING	
CRD$SCAN$NUMERIC	00000020-R	CRDVECS	MENTST	
CRD$SCREEN_PAUSE	00004496-R	CRDSTOP	CRDMMACT	CRDSTOP
			MENCNTLC	MENPROMPT
			MENTST	
CRD$SET_CTRL_C_FLAG	000012C4-R	CRDCTRLC	CRDCTRLC	
CRD$SET_CTRL_C_HANDLERS	00001294 R	CRDCTRLC	CRDCTRLC	CRDINIT
			MENCNTLC	

Symbol	Value	Defined By	Referenced By ...	
CRD$SET_UP_DIAGNOSTIC	00001114-R	CRDACT2	CRDACT2	CRDSTATE
CRD$START_AUTOSIZER	00001269-R	CRDACT2	CRDACT2	CRDSTATE
CRD$START_ERROR	00000BC5-R	CRDACT1	CRDACT1	CRDHSACT
			CRDSTATE	
CRD$STOP	00003E5C-R	CRDSTOP	CRDACT1	CRDACT2
			CRDHSACT	CRDMMACT
			CRDSTOP	CRDTQACT
			MENCNTLC	MENPROMPT
			MENTSTINI	
CRD$TURING_MACHINE	000039C0-R	CRDSTATE	CRDINTRFC	CRDSTATE
DS$ABORT	00010020	CRDSTATE	CRDACT1	CRDHSACT
DS$ASKADR	00010090	CRDSTATE		
DS$ASKDATA	00010080	CRDSTATE		
DS$ASKLGCL	00010098	CRDSTATE		
DS$ASKSTR	000100A0	CRDSTATE	CRDSTOP	MENCNTLC
			MENPROMPT	MENTST
DS$ASKVLD	00010088	CRDSTATE		
DS$ATTACH	000101A8	CRDSTATE		
DS$BGNSUB	00010030	CRDSTATE		
DS$BRANCH	000100A8	CRDSTATE		
DS$BREAK	00010058	CRDSTATE	CRDSTATE	CRDSTOP
			MENCNTLC	MENPROMPT
			MENTST	MENTURING
DS$CANWAIT	00010070	CRDSTATE		
DS$CHANNEL	00010180	CRDSTATE		
DS$CKLOOP	00010040	CRDSTATE		
DS$CLRVEC	00010168	CRDSTATE		
DS$CNTRLC	00010078	CRDSTATE		
DS$CVTREG	000100B0	CRDSTATE		
DS$ENDPASS	00010010	CRDSTATE		
DS$ENDSUB	00010038	CRDSTATE		
DS$ERRDEV	000100C8	CRDSTATE		
DS$ERRHARD	000100D0	CRDSTATE		
DS$ERRPREP	00010108	CRDSTATE		
DS$ERRSOFT	000100D8	CRDSTATE		
DS$ERRSYS	000100C0	CRDSTATE		
DS$ESCAPE	00010050	CRDSTATE		
DS$FREEDBGSYM	000101B8	CRDSTATE		
DS$GETBUF	00010120	CRDSTATE		
DS$GETTERM	000101C8	CRDSTATE	CRDSTOP	MENTST
DS$GPHARD	00010018	CRDSTATE		
DS$HELP	000101B0	CRDSTATE		
DS$INITSCB	00010170	CRDSTATE		
DS$INLOOP	00010048	CRDSTATE		
DS$LOAD	00010198	CRDSTATE	MENTSTINI	
DS$LOADPCS	000101C0	CRDSTATE		
DS$MAPDBGBLOCK	00010118	CRDSTATE		
DS$MEMSIZE	000101D8	CRDSTATE		
DS$MMOFF	00010158	CRDSTATE		
DS$MMON	00010150	CRDSTATE		

Symbol	Value	Defined By	Referenced By ...	
DS$PARSE	000100B8	CRDSTATE		
DS$PRINTB	000100E0	CRDSTATE		
DS$PRINTF	000100F0	CRDSTATE		
DS$PRINTS	000100F8	CRDSTATE		
DS$PRINTSIG	00010100	CRDSTATE		
DS$PRINTX	000100E8	CRDSTATE		
DS$PROBE	000101A0	CRDSTATE		
DS$RELBUF	00010128	CRDSTATE		
DS$SERVICE_DISPATCHER	000101D0	CRDSTATE		
DS$SETIPL	00010178	CRDSTATE		
DS$SETMAP	00010188	CRDSTATE		
DS$SETPRIEXV	00010110	CRDSTATE		
DS$SETVEC	00010160	CRDSTATE		
DS$SHOCHAN	00010190	CRDSTATE		
DS$SUMMARY	00010028	CRDSTATE		
DS$WAITMS	00010060	CRDSTATE		
DS$WAITUS	00010068	CRDSTATE		
HS$ADVANCE_DIAGNOSTIC	00002B27-R	CRDHSACT	CRDHSACT	MENTURING
HS$ADVANCE_GENERAL_COM	00002BE5-R	CRDHSACT	CRDHSACT	MENTURING
HS$CHANGE_TABLE	00002D99-R	CRDHSACT	CRDHSACT	MENTURING
HS$DIAGNOSTIC_ERROR	00002DE7-R	CRDHSACT	CRDHSACT	MENTURING
HS$DONE_GENERAL_COMS	00002C03-R	CRDHSACT	CRDHSACT	MENTURING
HS$INIT_HS	00002B08-R	CRDHSACT	CRDHSACT	MENTURING
HS$INTERNAL_ERROR	00002E66-R	CRDHSACT	CRDHSACT	MENTURING
HS$LOAD_ERROR	00002DAD-R	CRDHSACT	CRDHSACT	MENTURING
HS$LOAD_GENERAL_COM	00002BD7-R	CRDHSACT	CRDHSACT	MENTURING
HS$LOAD_LOAD_COM	00002C0D-R	CRDHSACT	CRDHSACT	MENTURING
HS$LOAD_SELECT_COM	00002CFA-R	CRDHSACT	CRDHSACT	MENTURING
HS$LOAD_SET_EVENT_COM	00002CA7-R	CRDHSACT	CRDHSACT	MENTURING
HS$LOAD_SET_FLAGS_COM	00002C54-R	CRDHSACT	CRDHSACT	MENTURING
HS$LOAD_START_COM	00002D4E-R	CRDHSACT	CRDHSACT	MENTURING
HS$PREPARATION_ERROR	00002E3A-R	CRDHSACT	CRDHSACT	MENTURING
HS$START_ERROR	00002DCA-R	CRDHSACT	CRDHSACT	MENTURING
LIB$ANALYZE_SDESC	0000F23C-R	LIB$ANALYZE_SDESC		
LIB$ANALYZE_SDESC_R2	0000F251-R	LIB$ANALYZE_SDESC	LIB$SPAWN	
LIB$SPAWN	0000F0CC-R	LIB$SPAWN	CRDMMACT	
LIB$_INVARG	00158234	LIB$MSGDEF	LIB$SPAWN	
LIB$_INVSTRDES	00158224	LIB$MSGDEF	LIB$ANALYZE_SDESC	
LIB$_NOCLI	0015837C	LIB$MSGDEF	LIB$SPAWN	
MEN$ALLOCATE_MEMORY	00006155-R	MENTSTINI	MENTSTINI	
MEN$BLDTSTQ_FNCTST_ALL	00006518-R	MENTSTQ	MENPROMPT	MENTSTQ
MEN$BLDTSTQ_FNCTST_DEV	000065A3-R	MENTSTQ	MENPROMPT	MENTSTQ
MEN$BUILD_DDB	00005DFB-R	MENTSTINI	CRDMMACT	MENTSTINI
MEN$CNTLCINIT_TBL_INIT	0000491C-R	MENCNTLC	MENCNTLC	MENTSTINI
MEN$CNTLC_MENU	00004A36-R	MENCNTLC	MENCNTLC	MENTSTINI
MEN$CNTLC_TBL_INIT	00004952-R	MENCNTLC	MENCNTLC	MENTSTINI
MEN$CVT_TIME_ASC_D	00005983-R	MENTST	MENTST	
MEN$DETERMINE_SUPPORT	00005F03-R	MENTSTINI	CRDMMACT	MENTSTINI
MEN$DUMP_TSTQ_HST	0000633D-R	MENTSTINI	MENPROMPT	MENTSTINI
MEN$FIND_SUPBLK	000062BF-R	MENTSTINI	MENTSTINI	

Symbol	Value	Defined By	Referenced By ...	
MEN$FNCTST_CMPLERR_STATUS	000056A4-R	MENTST	CRDTQACT	MENTST
MEN$FNCTST_TESTSTRT_TEXT	000057E7-R	MENTST	CRDHSACT	MENTST
			MENTSTINI	
MEN$FNDDDB_TCL	000053C2-R	MENPROMPT	MENPROMPT	
MEN$FNDDDB_TCL_SDN	00005371-R	MENPROMPT	MENPROMPT	
MEN$FTMTBL_INIT	00004BFC-R	MENPROMPT	MENPROMPT	MENTSTINI
MEN$FUNCTEST_MENU	00004D03-R	MENPROMPT	CRDMMACT	MENPROMPT
MEN$GA_DDB_HEAD	0000F090-R	MENTSTINI	MENPROMPT	MENTSTINI
			MENTSTQ	
MEN$GA_FTMTBL	0000EF08-R	MENPROMPT	MENPROMPT	
MEN$GA_FUNCSUP_DATBASE	0000F098-R	MENTSTINI	MENTSTINI	
MEN$GA_PREPTBL	0000EF40-R	MENPROMPT	MENPROMPT	
MEN$GA_TEST_QUEUE	0000F0A8-R	MENTSTINI	CRDTQACT	MENTST
			MENTSTINI	MENTSTQ
MEN$GA_TMTBL	0000EF28-R	MENPROMPT	MENPROMPT	MENTSTQ
MEN$GB_FNCTST_INIT	0000F0B4-R	MENTSTINI	MENCNTLC	MENPROMPT
			MENTSTINI	
MEN$GB_TEST_IN_PROGRESS	0000F0B5-R	MENTSTINI	MENPROMPT	MENTST
			MENTSTINI	
MEN$GENERATE_ONLINE_ATTACH	00000068-R	MENGOA	CRDMMACT	
MEN$GET_TSTCLA_NAME	000052C0-R	MENPROMPT	MENPROMPT	MENTST
MEN$GT_ASSISTANCE_MSG	000093B8-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_AUTOSIZER_BEGIN	00008CE5-R	CRDTEXT	CRDMMACT	CRDTEXT
MEN$GT_AUTOSIZER_END	00008D35-R	CRDTEXT	CRDMMACT	CRDTEXT
MEN$GT_CALL_FLDSRV	000094A1-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_CNTLCM_CHOICE	000090B0-R	CRDTEXT	CRDTEXT	MENCNTLC
MEN$GT_CNTLC_CONTINUING	0000911E-R	CRDTEXT	CRDTEXT	MENCNTLC
MEN$GT_CNTLC_MENU_PROMPT	000090F8-R	CRDTEXT	CRDTEXT	MENCNTLC
MEN$GT_CNTLC_MENU_SELECTION	00009039-R	CRDTEXT	CRDTEXT	MENCNTLC
MEN$GT_CNTLC_MENU_TITLE	0000900C-R	CRDTEXT	CRDTEXT	MENCNTLC
MEN$GT_CNTLC_SEL_CONTINUE	0000904D-R	CRDTEXT	CRDTEXT	MENCNTLC
MEN$GT_CNTLC_SEL_EXIT	00008DA1-R	CRDTEXT	CRDTEXT	MENCNTLC
MEN$GT_CNTLC_SEL_FTMENU	00009079-R	CRDTEXT	CRDTEXT	MENCNTLC
MEN$GT_CTRLC_FNCTST	00009258-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_1	000097D8-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_2	000098A9-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_3	00009914-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_4	000099DC-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_5	00009A5D-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_6	00009ADA-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_7	00009B9A-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_NAME	00009789-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_NULL	000097AC R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_DEVTSTPREP_TEXT	0000979C-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_END_OF_LIST	00009244-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FAILED_HDWR_NAME	000094E3-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_FAILED_ONL_AUTOSIZER	00009594-R	CRDTEXT	CRDMMACT	CRDTEXT
MEN$GT_FILE_DEFAULT	0000DB5C-R	MENTSTINI	MENTSTINI	
MEN$GT_FNCTST_CMPL_ERR	0000936B-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_FNCTST_CMPL_NOERR	00009315-R	CRDTEXT	CRDTEXT	MENTST

Symbol	Value	Defined By	Referenced By ...	
MEN$GT_FNCTST_INCMPL_ERR	0000933F-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_FNCTST_INCMPL_NOERR	00009395-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_FNCTST_TESTSTRT_TEXT	00009465-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_FTMENU_SELECTION	00008D91-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FTMPROMPT	00008EAF-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FTMSEL_EXIT	00008DA1-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FTMSEL_FNCTST_ALL	00008E4E-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FTMSEL_PREP	00008DB0-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FTMSEL_SYSTEM_HDWR	00008E12-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FTMSEL_TEST	00008DEA-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FTMTITLE	00008D63-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FTM_CHOICE	00008E6A-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_FUNCTEST_BEGIN	000092BC-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_INVOPERINPUT	000094BC-R	CRDTEXT	CRDSTOP	CRDTEXT
			MENCNTLC	MENPROMPT
			MENTST	
MEN$GT_MENFUNCSUP_FILE	0000DB52-R	MENTSTINI	CRDSTOP	MENTSTINI
MEN$GT_MENUTEST_ABORTED	000093EA-R	CRDTEXT	CRDTEXT	
MEN$GT_MENUTEST_BEGIN	00008C13-R	CRDTEXT	CRDMMACT	CRDTEXT
MEN$GT_NOALL_NONPGMEM	000096D1-R	CRDTEXT	CRDTEXT	MENTSTINI
MEN$GT_NOSUPPORT	00009239-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_NULL	000094A0-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_OPERATOR_ABORT	0000940B-R	CRDTEXT	CRDSTOP	CRDTEXT
MEN$GT_OPERATOR_STOP	00009438-R	CRDTEXT	CRDSTOP	CRDTEXT
MEN$GT_OPERINPUT_BUFFER	0000EE08-R	MENPROMPT	MENPROMPT	MENTST
MEN$GT_PREP_ERROR_TEXT	00009505-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_PREP_ERROR_TEXT_NO_USER	00009556-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_PRESS_RETURN_TM	00008FEE-R	CRDTEXT	CRDSTOP	CRDTEXT
MEN$GT_RUNTIME_TOTAL	000092EB-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_SCRMEDIA_HDWR	0000966F-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_SCRMEDIA_HDWR_END	0000968D-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_SCRMEDIA_MSG	000095C0-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_SCRMEDIA_PROMPT	00009695-R	CRDTEXT	CRDTEXT	MENTST
MEN$GT_SUPPORTED	0000922F-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_SUPPORT_FILE_LOAD_ERR	0000973B-R	CRDTEXT	CRDSTOP	CRDTEXT
MEN$GT_SUPPORT_FILE_NOT_FOUND	000096F3-R	CRDTEXT	CRDSTOP	CRDTEXT
MEN$GT_SYSHDWR_TITLE	000091B1-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_SYSTEM_HDWR_TEXT	0000921D-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TMENU_SELECTION	00008EE6-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TMENU_TSTALL	00008F07-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TMPROMPT	00008F99-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TMTITLE	00008ECC-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TM_CHOICE	00008F42-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TM_MSG	00008FB6-R	CRDTEXT	CRDSTOP	CRDTEXT
MEN$GT_TSTCLA_ALL	000091A7-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_CARD	00009193-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_COMM	0000916B-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_CPU	00009143-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_DISK	00009157-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_IOADAP	0000919D-R	CRDTEXT	CRDTEXT	MENPROMPT

Symbol	Value	Defined By	Referenced By ...	
MEN$GT_TSTCLA_MEMORY	0000914D-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_NULLNAME	000091A7-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_PRINTER	00009189-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_REAL	00009175-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_TAPE	00009161-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$GT_TSTCLA_TERM	0000917F-R	CRDTEXT	CRDTEXT	MENPROMPT
MEN$INIT_HS_STATE_TABLE	000065D8-R	MENTURING	MENTSTINI	MENTURING
MEN$INIT_MM_STATE_TABLE	000065C4-R	MENTURING	MENTSTINI	MENTURING
MEN$INIT_TQ_STATE_TABLE	000065CE-R	MENTURING	MENTSTINI	MENTURING
MEN$LOAD_FILE	000061BE-R	MENTSTINI	MENTSTINI	
MEN$MENU_CLEANUP	00006480-R	MENTSTINI	CRDSTOP	MENTSTINI
MEN$MENU_FNCTST_INIT	00005D14-R	MENTSTINI	CRDINTRFC	MENTSTINI
MEN$MENU_SUMMARY	00005B57-R	MENTST	MENTST	
MEN$MENU_TEST_INTERNAL_ERROR	0000370F-R	CRDMMACT	CRDHSACT	CRDMMACT
			CRDTQACT	MENTSTINI
MEN$PREPARATION_ERR_TEXT	000059F1-R	MENTST	CRDHSACT	MENTST
MEN$PREPTBL_INIT	00004C32-R	MENPROMPT	MENPROMPT	MENTSTINI
MEN$PRINT_ACTION_ROUTINE	00006AFB-R	MENTURING	MENTURING	
MEN$PRINT_DEVTSTPREP	00005404-R	MENPROMPT	MENPROMPT	
MEN$PRINT_FAILED_HDWR	00005840 R	MENTST	MENTST	
MEN$PRINT_RUNTIME_TOTAL	00005909-R	MENTST	MENPROMPT	MENTST
MEN$PRINT_SYSTEM_HDWR	000055AF-R	MENPROMPT	MENPROMPT	
MEN$SCAN_NUMERIC_ASC_D	000059DA-R	MENTST	MENPROMPT	MENTST
MEN$SCRATCH_MEDIA_MSG	00005A46-R	MENTST	MENPROMPT	MENTST
MEN$SPECIFY_SELECTNO	000060F4-R	MENTSTINI	CRDMMACT	MENTSTINI
MEN$TEST_MENU	000051C3-R	MENPROMPT	MENPROMPT	
MEN$TURING_MACHINE	00006786-R	MENTURING	CRDINTRFC	MENTURING
MM$AUTOSIZER_ERROR	000036FA-R	CRDMMACT	CRDMMACT	MENTURING
MM$BUILD_DDBS	00003673-R	CRDMMACT	CRDMMACT	MENTURING
MM$BUILD_HSTS	0000369A-R	CRDMMACT	CRDMMACT	MENTURING
MM$CHANGE_TABLE	000036C8-R	CRDMMACT	CRDMMACT	MENTURING
MM$DONE_INITS	000036AE-R	CRDMMACT	CRDMMACT	MENTURING
MM$INTERNAL_ERROR	000036DA-R	CRDMMACT	CRDMMACT	MENTURING
MM$LOAD_AUTOSIZER	000031B7-R	CRDMMACT	CRDMMACT	MENTURING
MM$MENU_PROMPT	000036D5-R	CRDMMACT	CRDMMACT	MENTURING
MM$PRINT_MENU	000036B4-R	CRDMMACT	CRDMMACT	MENTURING
MM$PRINT_MENU_HEADING	000031A0-R	CRDMMACT	CRDMMACT	MENTURING
MM$SET_FLAGS_AUTOSIZER	000032FA-R	CRDMMACT	CRDMMACT	MENTURING
MM$START_AUTOSIZER	00003520-R	CRDMMACT	CRDMMACT	MENTURING
SYS$ALLOC	00010238	CRDSTATE		
SYS$ASCTIM	00010248	CRDSTATE		
SYS$ASSIGN	00010250	CRDSTATE		
SYS$BINTIM	00010258	CRDSTATE		
SYS$CANCEL	00010260	CRDSTATE		
SYS$CANTIM	00010268	CRDSTATE		
SYS$CLI	7FFEDE18	SYS$P1_VECTOR	LIB$SPAWN	
SYS$CLOSE	000105B8	CRDSTATE	CRDINTRFC	CRDMMACT
			MENTSTINI	
SYS$CLREF	00010298	CRDSTATE		
SYS$CONNECT	000105C0	CRDSTATE	CRDMMACT	

Symbol	Value	Defined By	Referenced By ...		
SYS$DALLOC	000102D8	CRDSTATE			
SYS$DASSGN	000102E0	CRDSTATE			
SYS$DISCONNECT	000105D0	CRDSTATE			
SYS$ERASE	7FFEE1E0	SYS$P1_VECTOR	CRDINTRFC	CRDMMACT	
SYS$FAO	00010350	CRDSTATE	CRDINTRFC MENGCA	CRDMMACT	
SYS$FAOL	00010358	CRDSTATE			
SYS$GET	00010580	CRDSTATE	CRDMMACT		
SYS$GETCHN	000104C8	CRDSTATE			
SYS$GETTIM	00010378	CRDSTATE			
SYS$GTCHAN	00010380	CRDSTATE			
SYS$HIBER	00010380	CRDSTATE			
SYS$LKWSET	000103A0	CRDSTATE			
SYS$NUMTIM	000103B8	CRDSTATE			
SYS$OPEN	00010608	CRDSTATE	CRDINTRFC MENTSTINI	CRDMMACT	
SYS$QIO	000103C8	CRDSTATE			
SYS$QIOW	00010200	CRDSTATE			
SYS$READ	00010590	CRDSTATE			
SYS$READEF	000103D0	CRDSTATE			
SYS$SETAST	000103F8	CRDSTATE			
SYS$SETEF	00010400	CRDSTATE			
SYS$SETIMR	00010420	CRDSTATE			
SYS$SETPRI	00010428	CRDSTATE			
SYS$SETPRT	00010430	CRDSTATE			
SYS$SETRWN	00010438	CRDSTATE			
SYS$ULKPAG	00010460	CRDSTATE			
SYS$ULWSET	00010468	CRDSTATE			
SYS$UNWIND	00010470	CRDSTATE			
SYS$WAITFR	00010478	CRDSTATE			
SYS$WFLAND	00010488	CRDSTATE			
SYS$WFLOR	00010490	CRDSTATE			
SYS$WRITE	7FFEE1B0	SYS$P1_VECTOR	CRDINTRFC		
TQ$CHANGE_TABLE	0000471B-R	CRDTQACT	CRDTQACT	MENTURING	
TQ$DONE_TEST_QUEUE	00004728-R	CRDTQACT	CRDTQACT	MENTURING	
TQ$GET_DDB	00004685-R	CRDTQACT	CRDTQACT	MENTURING	
TQ$INIT_TQ	00004630-R	CRDTQACT	CRDTQACT	MENTURING	
TQ$INTERNAL_ERROR	00004742-R	CRDTQACT	CRDTQACT	MENTURING	

Symbols By Value

Value	Symbols...
00000000	R-CRD$AL_CRD_DISPATCH_VECTORS
00000004	R-CRD$EXIT_DS
00000008	R-CRD$PRINT
0000000C	R-CRD$GA_DS_FLAGS
00000010	R-CRD$GA_DS_CTRL_C_FIRST
00000014	R-CRD$GA_PTABLE
00000018	R-CRDEXEALONONPAGED
0000001C	R-CRD$GA_USER_ERROR_TEXT
00000020	R-CRD$SCAN$NUMERIC
00000024	R-CRDEXEDEANONPAGED
00000028	R-CRD$GA_DS_CTRL_C_SECOND
0000002C	R-CRD$GA_BEGIN
00000030	R-CRD$GB_CRD_DEBUG
00000034	R-CRDDSRCHECK_MENUTEST_OFF_SET
00000038	R-CRDDSRCHECK_AUTOTEST_OFF_SET
0000003C	R-CRDDSGA_PTDESC
00000068	R-MEN$GENERATE_ONLINE_ATTACH
0000087C	R-CRD$AUTOSIZER_ERROR
0000089C	R-CRD$DIAGNOSTIC_ERROR
000008F6	R-CRD$EXIT_CRD
000009BA	R-CRD$INTERNAL_ERROR
00000A34	R-CRD$LOAD_ERROR
00000A68	R-CRD$PREPARATION_ERROR
00000B27	R-CRD$PRINT_DDB
00000BC5	R-CRD$START_ERROR
00000BF8	R-CRD$ADVANCE_DIAGNOSTIC
00000C84	R-CRD$ADVANCE_GENERAL_COM
00000CC0	R-CRD$ASSIGN_PTABLE_PTRS
00000D10	R-CRD$AUTOSIZER_SET_FLAGS
00000D5D	R-CRD$DETERMINE_BASE_SYSTEM
00000E9F	R-CRD$DONE_GENERAL_COMS
00000EC9	R-CRD$LEVEL_4_PASSED
00000EE0	R-CRD$LOAD_AUTOSIZER
00000F27	R-CRD$LOAD_GENERAL_COM
00000F6C	R-CRD$LOAD_LOAD_COM
00000FC0	R-CRD$LOAD_SELECT_COM
00001017	R-CRD$LOAD_SET_EVENT_COM
0000106E	R-CRD$LOAD_SET_FLAGS_COM
000010C5	R-CRD$LOAD_START_COM
00001114	R-CRD$SET_UP_DIAGNOSTIC
00001269	R-CRD$START_AUTOSIZER
00001294	R-CRD$SET_CTRL_C_HANDLERS
000012A3	R-CRD$CLEAR_CTRL_C_HANDLERS
000012B0	R-CRD$ENABLE_CTRL_C
000012BA	R-CRD$DISABLE_CTRL_C
000012C4	R-CRD$SET_CTRL_C_FLAG
000012CC	R-CRD$CLEAR_CTRL_C_FLAG

```
Value                                   Symbols...
-----                                   ----------
000012D4      R-CRD$CHECK_CTRL_C
000012E0      R-CRD$BUILD_DDBS
00001F9F      R-CRD$DISPOSE
00001FD9      R-CRD$INIT_DDB_STATUS
00001FEC      R-CRD$NEW
00002054      R-CRD$INIT_GENERAL_COM_TABLE
0000205D      R-CRD$GET_GENERAL_COMMAND
00002081      R-CRD$PRINT_COMMAND_TABLE
000020E0      R-CRD$BUILD_SUPPORT_TABLE
00002A95      R-CRD$PRINT_HARDWARE_SUPPORT
00002B08      R-HS$INIT_HS
00002B27      R-HS$ADVANCE_DIAGNOSTIC
00002BD7      R-HS$LOAD_GENERAL_COM
00002BE5      R-HS$ADVANCE_GENERAL_COM
00002C03      R-HS$DONE_GENERAL_COMS
00002C0D      R-HS$LOAD_LOAD_COM
00002C54      R-HS$LOAD_SET_FLAGS_COM
00002CA7      R-HS$LOAD_SET_EVENT_COM
00002CFA      R-HS$LOAD_SELECT_COM
00002D4E      R-HS$LOAD_START_COM
00002D99      R-HS$CHANGE_TABLE
00002DAD      R-HS$LOAD_ERROR
00002DCA      R-HS$START_ERROR
00002DE7      R-HS$DIAGNOSTIC_ERROR
00002E3A      R-HS$PREPARATION_ERROR
00002E66      R-HS$INTERNAL_ERROR
00002E85      R-CRD$PRINT_DS_COMMAND
00002EA8      R-CRD$CRD_INITIALIZATION
00002F03      R-CRD$COMMON_INITIALIZATIONS
00002F70      R-CRD$DS_INTERFACE
000030B3      R-CRD$LOG_START
00003154      R-CRD$LOG_STOP
00003172      R-CRD$LOG_WRITE
000031A0      R-MM$PRINT_MENU_HEADING
000031B7      R-MM$LOAD_AUTOSIZER
000032FA      R-MM$SET_FLAGS_AUTOSIZER
00003520      R-MM$START_AUTOSIZER
00003673      R-MM$BUILD_DDBS
0000369A      R-MM$BUILD_HSTS
000036AE      R-MM$DONE_INITS
000036B4      R-MM$PRINT_MENU
000036C8      R-MM$CHANGE_TABLE
000036D5      R-MM$MENU_PROMPT
000036DA      R-MM$INTERNAL_ERROR
000036FA      R-MM$AUTOSIZER_ERROR
0000370F      R-MEN$MENU_TEST_INTERNAL_ERROR
000037E8      R-CRD$FIND_PTABLE
00003835      R-CRD$GET_DEVICE_NAME
00003865      R-CRD$PRINT_PTABLES
00003938      R-CRD$INIT_STATE_TABLE
```

Value	Symbols...
0000394F	R-CRD$GET_STATE
00003988	R-CRD$GET_ACTION_ROUTINE
000039C0	R-CRD$TURING_MACHINE
00003C31	R-CRD$PRINT_ACTION_ROUTINE
00003E5C	R-CRD$STOP
00004174	R-CRD$CONTROL_C_HANDLER
0000418A	R-CRD$AUTO_SUMMARY
00004496	R-CRD$SCREEN_PAUSE
00004630	R-TQ$INIT_TQ
00004685	R-TQ$GET_DDB
0000471B	R-TQ$CHANGE_TABLE
00004728	R-TQ$DONE_TEST_QUEUE
00004742	R-TQ$INTERNAL_ERROR
00004764	R-CRD$PRINT_TYPECODE_NAME
0000491C	R-MEN$CNTLCINIT_TBL_INIT
00004952	R-MEN$CNTLC_TBL_INIT
00004A36	R-MEN$CNTLC_MENU
00004BFC	R-MEN$FTMTBL_INIT
00004C32	R-MEN$PREPTBL_INIT
00004D03	R-MEN$FUNCTEST_MENU
000051C3	R-MEN$TEST_MENU
000052C0	R-MEN$GET_TSTCLA_NAME
00005371	R-MEN$FNDDDB_TCL_SDN
000053C2	R-MEN$FNDDDB_TCL
00005404	R-MEN$PRINT_DEVTSTPREP
000055AF	R-MEN$PRINT_SYSTEM_HDWR
000056A4	R-MEN$FNCTST_CMPLERR_STATUS
000057E7	R-MEN$FNCTST_TESTSTRT_TEXT
00005840	R-MEN$PRINT_FAILED_HDWR
00005909	R-MEN$PRINT_RUNTIME_TOTAL
00005983	R-MEN$CVT_TIME_ASC_D
000059DA	R-MEN$SCAN_NUMERIC_ASC_D
000059F1	R-MEN$PREPARATION_ERR_TEXT
00005A46	R-MEN$SCRATCH_MEDIA_MSG
00005B57	R-MEN$MENU_SUMMARY
00005D14	R-MEN$MENU_FNCTST_INIT
00005DFB	R-MEN$BUILD_DDB
00005F03	R-MEN$DETERMINE_SUPPORT
000060F4	R-MEN$SPECIFY_SELECTNO
00006155	R-MEN$ALLOCATE_MEMORY
000061BE	R-MEN$LOAD_FILE
000062BF	R-MEN$FIND_SUPBLK
0000633D	R-MEN$DUMP_TSTQ_HST
00006480	R-MEN$MENU_CLEANUP
00006518	R-MEN$BLDTSTQ_FNCTST_ALL
000065A3	R-MEN$BLDTSTQ_FNCTST_DEV
000065C4	R-MEN$INIT_MM_STATE_TABLE
000065CE	R-MEN$INIT_TQ_STATE_TABLE
000065D8	R-MEN$INIT_HS_STATE_TABLE
00006786	R-MEN$TURING_MACHINE

```
Value                                   Symbols...
-----                                   ----------
00006AFB      R-MEN$PRINT_ACTION_ROUTINE
00007B54      R-CRD$GA_GENERAL_COMMANDS
000084B6      R-CRD$GT_DS_LOAD_COM
000084BC      R-CRD$GT_DS_SET_FLAGS_COM
000084C7      R-CRD$GT_DS_SET_EVENT_COM
000084D8      R-CRD$GT_DS_SELECT_COM
000084E0      R-CRD$GT_DS_START_COM
00008690      R-CRD$GT_SPACE_AND_SPACE
00008696      R-CRD$GT_NEW_LINE
00008699      R-CRD$GT_TAB
0000869C      R-CRD$GT_SLASH
0000869E      R-CRD$GT_UNKNOWN
000086A3      R CRD$GT_2_TABS
000086A8      R-CRD$GT_MENU
000086AD      R-CRD$GT_AUTO
000086B2      R-CRD$GT_TESTING_DEVICE
000086BA      R-CRD$GT_TESTING_STARTED
000086D8      R-CRD$GT_RUNTIME
000086E6      R-CRD$GT_FAIL
000086F4      R-CRD$GT_PASS
00008702      R-CRD$GT_CRD_NO_ERRORS
00008724      R-CRD$GT_CRD_ERRORS_COMPLETE
00008743      R-CRD$GT_CRD_BAD_DEVICE
0000877A      R-CRD$GT_CRD_INCOMPLETE
000087BD      R-CRD$GT_CRD_DEV_UNTESTED
00008810      R-CRD$GT_CRD_EVSBA_ERROR
0000883A      R-CRD$GT_CRD_EVSBB_ERROR
00008864      R-CRD$GT_CRD_FATAL_ERROR
0000888E      R-CRD$GT_INTERNAL_ERROR_ABORT
000088B8      R-CRD$GT_CONTROL_C_ABORT
000088EB      R-CRD$GT_CRD_END_MESSAGE
00008912      R-CRD$GT_EXIT_TO_CONSOLE
0000894F      R-CRD$GT_EXIT_TO_CLI
00008989      R-CRD$GT_PRESS_RETURN
000089A2      R-CRD$GT_PRESS_RETURN_MM
000089C0      R-CRD$GT_INV_BASE_SYSTEM
000089F7      R-CRD$GT_AUTOSIZER_ERROR
00008A43      R-CRD$GT_DEV_UNAVAILABLE
00008A5C      R-CRD$GT_DEV_UNAVAILABLE_2
00008A73      R-CRD$GT_CANNOT_LOAD_DIAG
00008A93      R-CRD$GT_START_ERROR
00008ABB      R-CRD$GT_PREP_ERROR_SPACES
00008AC5      R-CRD$GT_INTERNAL_ERROR
00008AF5      R-CRD$GT_DS_COMMAND_OUT
00008B13      R-CRD$GT_SAME_LINE_MESSAGE
00008B14      R-CRD$GT_NEW_LINE_MESSAGE
00008B17      R-CRD$GT_STAR_LINE_PREFIX_MESSAGE
00008B1D      R-CRD$GT_STAR_LINE_SUFFIX_MESSAGE
00008B21      R-CRD$GT_SUMMARY_TITLE
00008B35      R-CRD$GT_ALL_FAILURES
```

```
Value                                   Symbols...
-----                                   ----------
00008B63      R-CRD$GT_ALL_PASSES
00008B91      R-CRD$GT_ALL_INC_HDWR_PREP
00008BD7      R-CRD$GT_ALL_OTHERS
00008BFD      R-CRD$GT_INIT_ALL_PASSES
00008C13      R-MEN$GT_MENUTEST_BEGIN
00008CE5      R-MEN$GT_AUTOSIZER_BEGIN
00008D35      R-MEN$GT_AUTOSIZER_END
00008D63      R-MEN$GT_FTMTITLE
00008D91      R-MEN$GT_FTMENU_SELECTION
00008DA1      R-MEN$GT_CNTLC_SEL_EXIT               R-MEN$GT_FTMSEL_EXIT
00008DB0      R-MEN$GT_FTMSEL_PREP
00008DEA      R-MEN$GT_FTMSEL_TEST
00008E12      R-MEN$GT_FTMSEL_SYSTEM_HDWR
00008E4E      R-MEN$GT_FTMSEL_FNCTST_ALL
00008E6A      R-MEN$GT_FTM_CHOICE
00008EAF      R-MEN$GT_FTMPROMPT
00008ECC      R-MEN$GT_TMTITLE
00008EE6      R-MEN$GT_TMENU_SELECTION
00008F07      R-MEN$GT_TMENU_TSTALL
00008F42      R-MEN$GT_TM_CHOICE
00008F99      R-MEN$GT_TMPROMPT
00008FB6      R-MEN$GT_TM_MSG
00008FEE      R-MEN$GT_PRESS_RETURN_TM
0000900C      R-MEN$GT_CNTLC_MENU_TITLE
00009039      R-MEN$GT_CNTLC_MENU_SELECTION
0000904D      R-MEN$GT_CNTLC_SEL_CONTINUE
00009079      R-MEN$GT_CNTLC_SEL_FTMENU
000090B0      R-MEN$GT_CNTLCM_CHOICE
000090F8      R-MEN$GT_CNTLC_MENU_PROMPT
0000911E      R-MEN$GT_CNTLC_CONTINUING
00009143      R-MEN$GT_TSTCLA_CPU
0000914D      R-MEN$GT_TSTCLA_MEMORY
00009157      R-MEN$GT_TSTCLA_DISK
00009161      R-MEN$GT_TSTCLA_TAPE
0000916B      R-MEN$GT_TSTCLA_COMM
00009175      R-MEN$GT_TSTCLA_REAL
0000917F      R-MEN$GT_TSTCLA_TERM
00009189      R-MEN$GT_TSTCLA_PRINTER
00009193      R-MEN$GT_TSTCLA_CARD
0000919D      R-MEN$GT_TSTCLA_IOADAP
000091A7      R-MEN$GT_TSTCLA_ALL                   R-MEN$GT_TSTCLA_NULLNAME
000091B1      R-MEN$GT_SYSHDWR_TITLE
0000921D      R-MEN$GT_SYSTEM_HDWR_TEXT
0000922F      R-MEN$GT_SUPPORTED
00009239      R-MEN$GT_NOSUPPORT
00009244      R-MEN$GT_END_OF_LIST
00009258      R-MEN$GT_CTRLC_FNCTST
000092BC      R-MEN$GT_FUNCTEST_BEGIN
000092EB      R-MEN$GT_RUNTIME_TOTAL
00009315      R-MEN$GT_FNCTST_CMPL_NOERR
```

```
Value                                   Symbols...
-----                                   ----------
0000933F      R-MEN$GT_FNCTST_INCMPL_ERR
0000936B      R-MEN$GT_FNCTST_CMPL_ERR
00009395      R-MEN$GT_FNCTST_INCMPL_NOERR
000093B8      R-MEN$GT_ASSISTANCE_MSG
000093EA      R-MEN$GT_MENUTEST_ABORTED
0000940B      R-MEN$GT_OPERATOR_ABORT
00009438      R-MEN$GT_OPERATOR_STOP
00009465      R-MEN$GT_FNCTST_TESTSTRT_TEXT
000094A0      R-MEN$GT_NULL
000094A1      R-MEN$GT_CALL_FLDSRV
000094BC      R-MEN$GT_INVOPERINPUT
000094E3      R-MEN$GT_FAILED_HDWR_NAME
00009505      R-MEN$GT_PREP_ERROR_TEXT
00009556      R-MEN$GT_PREP_ERROR_TEXT_NO_USER
00009594      R-MEN$GT_FAILED_ONL_AUTOSIZER
000095C0      R-MEN$GT_SCRMEDIA_MSG
0000966F      R-MEN$GT_SCRMEDIA_HDWR
0000968D      R-MEN$GT_SCRMEDIA_HDWR_END
00009695      R-MEN$GT_SCRMEDIA_PROMPT
000096D1      R-MEN$GT_NOALL_NONPGMEM
000096F3      R MEN$GT_SUPPORT_FILE_NOT_FOUND
0000973B      R-MEN$GT_SUPPORT_FILE_LOAD_ERR
00009789      R-MEN$GT_DEVTSTPREP_NAME
0000979C      R-MEN$GT_DEVTSTPREP_TEXT
000097AC      R-MEN$GT_DEVTSTPREP_NULL
000097D8      R-MEN$GT_DEVTSTPREP_1
000098A9      R MEN$GT_DEVTSTPREP_2
00009914      R-MEN$GT_DEVTSTPREP_3
000099DC      R-MEN$GT_DEVTSTPREP_4
00009A5D      R-MEN$GT_DEVTSTPREP_5
00009ADA      R MEN$GT_DEVTSTPREP_6
00009B9A      R MEN$GT_DEVTSTPREP_7
00009BDD      R CRD$GT_AUTOSIZER_NAME
00009BE3      R-CRD$GT_AUTOSIZER_SET_FLAGS
00009E6C      R-CRD$GW_STATE_TABLE
0000BBE8      R-CRD$GAW_MM_STATE_TABLE
0000C7C8      R-CRD$GAW_TQ_STATE_TABLE
0000CDB8      R-CRD$GAW_HS_STATE_TABLE
0000DB52      R MEN$GT_MENFUNCSUP_FILE
0000DB5C      R MEN$GT_FILE_DEFAULT
0000E65C      R-CRD$GA_CURRENT_DIAGNOSTIC
0000E660      R CRD$GA_FIRST_DIAGNOSTIC
0000E664      R CRD$GL_CURRENT_GENERAL_COM
0000E668      R-CRD$GB_BASE_SYSTEM
0000E66C      R-CRD$GA_HDWR_SPRT_TABLE
0000E670      R-CRD$AL_IDC_HDWR_SPRT_TABLE
0000E710      R-CRD$AL_LESI_HDWR_SPRT_TABLE
0000E770      R-CRD$AL_UDA_0_HDWR_SPRT_TABLE
0000E7F0      R CRD$AL_UDA_1_HDWR_SPRT_TABLE
0000E890      R-CRD$AL_UDA_2_HDWR_SPRT_TABLE
```

```
Value                                   Symbols...
-----                                   ----------
0000E910      R-CRD$AL_UDA_3_HDWR_SPRT_TABLE
0000E9B0      R-CRD$GT_WHICH_CRD
0000E9B4      R-CRD$GL_SAVED_DSA_FLAGS
0000EA4C      R-CRD$GB_USER_PROMPT_OKAY
0000EAE8      R-CRD$GA_CRD_MESSAGE_BUFFER
0000EB84      R-CRD$GA_REC_BUF
0000EBF8      R-CRD$GA_FILE_FAB
0000EC48      R-CRD$GA_FILE_RAB
0000ED6C      R-CRD$GL_CURRENT_TYPECODE
0000ED70      R-CRD$GL_CURRENT_STATE
0000ED74      R-CRD$GL_CURRENT_ACTION
0000ED7C      R-CRD$GB_AUTO_TO_MENU_OFF
0000ED80      R-CRD$GL_CRD_DEBUG_VECTOR
0000EE08      R-MEN$GT_OPERINPUT_BUFFER
0000EF08      R-MEN$GA_FTMTBL
0000EF28      R-MEN$GA_TMTBL
0000EF40      R-MEN$GA_PREPTBL
0000EF6C      R-CRD$GL_PREVIOUS_STATE
0000EF70      R-CRD$GL_MM_CURRENT_STATE
0000EF74      R-CRD$GL_TQ_CURRENT_STATE
0000EF78      R-CRD$GL_HS_CURRENT_STATE
0000EF7C      R-CRD$GL_TQ_CURRENT_TQ_ENTRY
0000EF80      R-CRD$GA_HS_CURRENT_DDB_PTR
0000EF84      R-CRD$GA_HS_CURRENT_HSB_PTR
0000EF88      R-CRD$GL_HS_CURRENT_HSB_NUMB
0000EF8C      R-CRD$GB_CURRENT_STATE_TABLE
0000F090      R-MEN$GA_DDB_HEAD
0000F098      R-MEN$GA_FUNCSUP_DATBASE
0000F0A8      R-MEN$GA_TEST_QUEUE
0000F0B4      R-MEN$GB_FNCTST_INIT
0000F0B5      R-MEN$GB_TEST_IN_PROGRESS
0000F0CC      R-LIB$SPAWN
0000F23C      R-LIB$ANALYZE_SDESC
0000F251      R-LIB$ANALYZE_SDESC_R2
00010010        DS$ENDPASS
00010018        DS$GPHARD
00010020        DS$ABORT
00010028        DS$SUMMARY
00010030        DS$BGNSUB
00010038        DS$ENDSUB
00010040        DS$CKLOOP
00010048        DS$INLOOP
00010050        DS$ESCAPE
00010058        DS$BREAK
00010060        DS$WAITMS
00010068        DS$WAITUS
00010070        DS$CANWAIT
00010078        DS$CNTRLC
00010080        DS$ASKDATA
00010088        DS$ASKVLD
```

Value	Symbols...
00010090	DS$ASKADR
00010098	DS$ASKLGCL
000100A0	DS$ASKSTR
000100A8	DS$BRANCH
000100B0	DS$CVTREG
000100B8	DS$PARSE
000100C0	DS$ERRSYS
000100C8	DS$ERRDEV
000100D0	DS$ERRHARD
000100D8	DS$ERRSOFT
000100E0	DS$PRINTB
000100E8	DS$PRINTX
000100F0	DS$PRINTF
000100F8	DS$PRINTS
00010100	DS$PRINTSIG
00010108	DS$ERRPREP
00010110	DS$SETPRIEXV
00010118	DS$MAPDBGBLOCK
00010120	DS$GETBUF
00010128	DS$RELBUF
00010150	DS$MMON
00010158	DS$MMOFF
00010160	DS$SETVEC
00010168	DS$CLRVEC
00010170	DS$INITSCB
00010178	DS$SETIPL
00010180	DS$CHANNEL
00010188	DS$SETMAP
00010190	DS$SHOCHAN
00010198	DS$LOAD
000101A0	DS$PROBE
000101A8	DS$ATTACH
000101B0	DS$HELP
000101B8	DS$FREEDBGSYM
000101C0	DS$LOADPCS
000101C8	DS$GETTERM
000101D0	DS$SERVICE_DISPATCHER
000101D8	DS$MEMSIZE
00010200	SYS$QIOW
00010238	SYS$ALLOC
00010248	SYS$ASCTIM
00010250	SYS$ASSIGN
00010258	SYS$BINTIM
00010260	SYS$CANCEL
00010268	SYS$CANTIM
00010298	SYS$CLREF
000102D8	SYS$DALLOC
000102E0	SYS$DASSGN
00010350	SYS$FAO
00010358	SYS$FAOL

Value	Symbols...		
00010378	SYS$GETTIM		
00010380	SYS$GTCHAN		SYS$HIBER
000103A0	SYS$LKWSET		
000103B8	SYS$NUMTIM		
000103C8	SYS$QIO		
000103D0	SYS$READEF		
000103F8	SYS$SETAST		
00010400	SYS$SETEF		
00010420	SYS$SETIMR		
00010428	SYS$SETPRI		
00010430	SYS$SETPRT		
00010438	SYS$SETRWN		
00010460	SYS$ULKPAG		
00010468	SYS$ULWSET		
00010470	SYS$UNWIND		
00010478	SYS$WAITFR		
00010488	SYS$WFLAND		
00010490	SYS$WFLOR		
000104C8	SYS$GETCHN		
00010580	SYS$GET		
00010590	SYS$READ		
000105B8	SYS$CLOSE		
000105C0	SYS$CONNECT		
000105D0	SYS$DISCONNECT		
00010608	SYS$OPEN		
00158224	LIB$_INVSTRDES		
00158234	LIB$_INVARG		
0015837C	LIB$_NOCLI		
7FFEDE18	SYS$CLI		
7FFEE1B0	SYS$WRITE		
7FFEE1E0	SYS$ERASE		

Key for special characters above:

*	- Undefined
U	- Universal
R	- Relocatable
X	- External

Image Synopsis

```
Virtual memory allocated:                 00000000 0000F3FF 0000F400 (62464. bytes, 122. pages)
Stack size:                                      0. pages
Image binary virtual block limits:               1.      122. (  122. blocks)
Image name and identification:            ENSAB NONE
Number of files:                                26.
Number of modules:                              30.
Number of program sections:                     12.
Number of global symbols:                      431.
Number of cross references:                   1316.
Number of image sections:                        1.
Image type:                               SYSTEM.
Map format:                               FULL WITH CROSS REFERENCE in file DISK$DIAGPACK03:[DS.CRD.V020]ENSAB.MAP;1
Estimated map length:                     155. blocks
```

Link Run Statistics

Performance Indicators	Page Faults	CPU Time	Elapsed Time
Command processing:	290	00:00:02.01	00:00:02.84
Pass 1:	399	00:00:04.60	00:00:11.88
Allocation/Relocation:	52	00:00:00.28	00:00:00.92
Pass 2:	193	00:00:04.49	00:00:12.68
Map data after object module synopsis:	9	00:00:02.27	00:00:02.57
Symbol table output:	0	00:00:00.00	00:00:00.13
Total run values:	943	00:00:13.65	00:00:31.02

```
Using a working set limited to 2048 pages and 112 pages of data storage (excluding image)

Total number object records read (both passes):   8990
    of which 53 were in libraries and 3474 were DEBUG data records containing 773100 bytes

Number of modules extracted explicitly              = 0
    with 5 extracted to resolve undefined symbols

0 library searches were for symbols not in the library searched

A total of 0 global symbol table records was written

LINK BUILD_CRD.OPT/OPTIONS/MAP=ENSAB.MAP/CROSS/FULL/NOSYSSHR/SYSTEM=%X0000/CONTIGUOUS/EXE=ENSAB.EXE
CRDACT1.OBJ -
,CRDACT2.OBJ -
,CRDCTRLC.OBJ -
,CRDDDB.OBJ -
,CRDGENCOM.OBJ -
,CRDHDWSUP.OBJ -
,CRDHSACT.OBJ -
,CRDINIT.OBJ -
,CRDINTRFC.OBJ -
```

```
,CRDTEXT.OBJ -
,CRDMMACT.OBJ -
,CRDPTABLE.OBJ -
,CRDSTATE.OBJ -
,CRDSTOP.OBJ -
,CRDTQACT.OBJ -
,CRDTYPCOD.OBJ -
,CRDVECS.OBJ -
,MENCNTLC.OBJ -
,MENGOA.OBJ -
,MENPROMPT.OBJ -
,MENSTATE.OBJ -
,MENTST.OBJ
,MENTSTINI.OBJ
,MENTSTQ.OBJ -
,MENTURING.OBJ
```

```
;    0001  0
;    0002  0 %TITLE  'CRD MENU - Functional Test Support'
;    0003  0 MODULE EVLAA (
;    0004  0   IDENT = 'V020'
;    0005  0   ) =
;    0006  0 !
;    0007  0 ! COPYRIGHT (C)  1985
;    0008  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0009  0 !
;    0010  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0011  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0012  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0013  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0014  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0015  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0016  0 ! REMAIN IN DEC.
;    0017  0 !
;    0018  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0019  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0020  0 ! CORPORATION.
;    0021  0 !
;    0022  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0023  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0024  0 !
```

```
;    0025  0 !++
;    0026  0 ! FACILITY:
;    0027  0 !
;    0028  0 ! ABSTRACT:
;    0029  0 !+
;    0030  0 !   This source causes a device support data base to be built.
;    0031  0 !
;    0032  0 ! ENVIRONMENT:
;    0033  0 !
;    0034  0 ! AUTHOR:
;    0035  0 !
;    0036  0 !  John Ciukaj May-1982 Version 1
;    0037  0 !
;    0038  0 !  Compatible with CRD MACRO Package
;    0039  0 !  Version 1.10 (ENSAB.EXE CRD Loadable Image)
;    0040  0 !  which incorporated the initial  release of VAX-11/730,725
;    0041  0 !  MENU TEST.
;    0042  0 !
;    0043  0 ! Modified by:
;    0044  0 !
;    0045  0 ! [01]  John Ciukaj Jan. 1983
;    0046  0 !  - Corrected version 1.0 description of this
;    0047  0 !    module from referencing CRD Macro Pkg. version  2.0
;    0048  0 !    to version 1.1
;    0049  0 !    Added device support:
;    0050  0 !    RA80, RA81, RA60, DZ32, DUP11, TU80, RK711
;    0051  0 !
;    0052  0 ! [02] Scott Cohen Version 1.3 13-Jul-1983
;    0053  0 !  - Added device support:
;    0054  0 !    RC25, RCF25
;    0055  0 !
;    0056  0 ! [03] Scott Cohen Release 15 8-Sep-1983
;    0057  0 !  - Added device support:
;    0058  0 !    VS125, TS05
;    0059  0 !  - Modified device support:
;    0060  0 !    RC25, RCF25
;    0061  0 !
;    0062  0 ! [04] Ron Parker Release 18 18-JUN-1984
;    0063  0 !  - Enhanced definition explanations
;    0064  0 !  - Added device support:
;    0065  0 !    VS100, UNA11
;    0066  0 !  - Commented out VS125 code
;    0067  0 !
;    0068  0 ! [05] Ron Parker Release 20 10-OCT-1984
;    0069  0 !  - Added RUX50  devices, RX31,RX32,RX50,Rx180
;    0070  0 !
;    0071  0 ! [06] Ron Parker 25-JAN-1985
;    0072  0 !  - Added Support block diagnostic level field to
;    0073  0 !    determine online versus offline items.
;    0074  0 !--
;    0075  0
;    0076  0
;    0077  0
```

```
;    0078  0 %SBTTL 'Libraries'
;    0079  1 BEGIN
;    0080  1
;    0081  1
;    0082  1 LIBRARY '$CRD';
;    0083  1 LIBRARY 'SYS$LIBRARY:LIB';
;    0084  1
;    0085  1 %SBTTL 'Literals'
;    0086  1
;    0087  1 !
;    0088  1 $CRD_Literals;
;    0089  1 $MEN_Literals;
;    0090  1 !
;    0091  1
;    0092  1 %SBTTL 'Notes'
;    0093  1
;    0094  1 ! The Diagnostic level word indicates whether or not the current
;    0095  1 ! block being defined contains offline or online diagnostics respectivily.
;    0096  1 ! This means level 3 is offline, and 2R is online. Level 2 diagnostics must
;    0097  1 ! be put in one or both catogories, level 3 and/or 2R. Level 2 will be
;    0098  1 ! interperted as 2R, so be carefull.
;    0099  1 !
;    0100  1 ! Support Block Status Bits.  $MEN_SupBlk_FLD defines the bit field.
;    0101  1 !    Status_supblk<0,1,0> = Support Block Invalid bit.
;    0102  1 ! Specifies whether CRD MENU considers the support block invalid.
;    0103  1 !  Valid   = 0 or False, Invalid = non-0 or True
```

```
;    0104  1 %SBTTL 'Device Data structure'
;    0105  1 !+
;    0106  1 !   The data structure consists of a pointer block (MEN$GA_FUNC_DATBASE) at
;    0107  1 !   the top of the structure. Next the Support blocks which may have one or
;    0108  1 !   more Test_configuration_blocks following. The test_configuration_blocks
;    0109  1 !   have one or more concatenated diagnostic_blocks following them.
;    0110  1 !
;    0111  1 !   MEN$GA_FUNCSUP_DATBASE block:
;    0112  1 !    .LONG Number of bytes in the databasse
;    0113  1 !    .LONG Pointer to the start of the database, first support_block
;    0114  1 !
;    0115  1 !   From this point on the database is one contiguous block consisting of:
;    0116  1 !
;    0117  1 !    Support_block:
;    0118  1 !    .LONG Number of longwords in block
;    0119  1 !    .ASCIC  The device name string (up to 11 char)
;    0120  1 !    .BYTE Test_catogory, see $MEN_literals in CRD.R32 for list
;    0121  1 !    .BYTE Test_class,    see $MEN_literals in CRD.R32 for list
;    0122  1 !    .BYTE Number of test_configuration_blocks in this support_block
;    0123  1 !    .WORD Support_indicator, TRUE/FALSE
;    0124  1 ! .WORD Diagnostic level, two characters
;    0125  1 !
;    0126  1 !   Test_configuration_block:
;    0127  1 !       .BYTE Device_test_preparation_code, see $MEN_literals in CRD.R32
;    0128  1 !       .BYTE Number of concatenated Diagnostic blocks
;    0129  1 !       .WORD Test configuration block status bits, see $MEN_TSTCNFG_FLD
;    0130  1 !
;    0131  1 !   Diagnostic_block:
;    0132  1 !    .WORD Length of block
;    0133  1 !    .ASCIC \Diagnostic_name\
;    0134  1 !    .ASCIC \Set_flags_args\
;    0135  1 !    .ASCIC \Set_event_args\
;    0136  1 !    .ASCIC \Start_qualifiers\
;    0137  1 !    .ASCIC \Test_start_text\
;    0138  1 !    .ASCIC  \Run_time\
;    0139  1 !
;    0140  1 !   Example structure in PDL
;    0141  1 !
;    0142  1 !   Begin_database_block
;    0143  1 !    Support_block
;    0144  1 !     Test_cnfg_block
;    0145  1 !          Diagnostic_block
;    0146  1 !                .
;    0147  1 !                .
;    0148  1 !          Diagnostic_block
;    0149  1 !    Support_block
;    0150  1 !     Test_cnfg_block
;    0151  1 !          Diagnostic_block
;    0152  1 !    Support_block
;    0153  1 !     Test_cnfg_block
;    0154  1 !          Diagnostic_block
;    0155  1 !   End_database_block
;    0156  1 !-
```

```
;    0157  1 %SBTTL 'Data Structure building MACROs'
;    0158  1
;    0159  1 KEYWORDMACRO
; M 0160  1  SUPPORT_DBASE_ID ( Version ) =
; M 0161  1
; M 0162  1     Psect plit=%Name('$$$$');
; M 0163  1     GLOBAL BIND
; M 0164  1     %NAME('CRD$GA_Support_DBase_ID')=
; M 0165  1
; M 0166  1     Plit Byte (
; M 0167  1     %IF %CHARCOUNT(Version) GTR 8
; M 0168  1     %THEN
; M 0169  1         %ERRORMACRO ('Version number character count exceeded')
; M 0170  1     %FI
; M 0171  1     byte (%ASCIC %STRING(%exactstring(19, %C' ', 'Version = ', Version)))
;    0172  1     )%;
;    0173  1
;    0174  1 KEYWORDMACRO
;    0175  1
;    0176  1 !+
;    0177  1 !    This macro builds a SUPPORT block header
;    0178  1 !-
;    0179  1
;    0180  1      INIT_SUPBLK_HEAD (   Device_Name,
;    0181  1       Test_Category,
;    0182  1       Test_Class,
;    0183  1       Testcnfg_No,
;    0184  1       Status_Supblk_Invalid = False,
;    0185  1       Diagnostic_Level
; M 0186  1      ) =
; M 0187  1
; M 0188  1   Psect plit=%Name('$$',Device_Name);
; M 0189  1   GLOBAL BIND
; M 0190  1   %NAME('CRD$GA_Supblk_L',
; M 0191  1    %STRING(%exactstring(2,%C' ',Diagnostic_Level)),
; M 0192  1    '_',
; M 0193  1    Device_Name)=
; M 0194  1
; M 0195  1     Plit Byte (
; M 0196  1
; M 0197  1       byte (%ASCIC %STRING(%exactstring(11,%c' ',Device_Name))),
; M 0198  1       byte (Test_Category),
; M 0199  1       byte (Test_Class),
; M 0200  1       byte (Testcnfg_No),
; M 0201  1   word ( (Status_Supblk_Invalid ^ 0 ) AND 1 ),
; M 0202  1   word ( %ASCII %STRING(%EXACTSTRING(2, %C' ',Diagnostic_Level)))! [06]
;    0203  1   %,
```

```
;    0204  1 !+
;    0205  1 !   This macro builds a TEST_CONFIGURATION block
;    0206  1 !-
;    0207  1
;    0208  1     INIT_TSTCNFG_BLK (  DevTstPrep_Code,
;    0209  1    Diagblk_No,
; M  0210  1    Status_Scratch_Media = False ) =
; M  0211  1
; M  0212  1  byte (DevTstPrep_Code),
; M  0213  1  byte (Diagblk_no),
; M  0214  1
; M  0215  1  !+
; M  0216  1  ! Test Configuration Block Status bits.  $MEN_TstCnfg_FLD defines
; M  0217  1  ! the bit field. The scratch media bit indicates that the hardware
; M  0218  1  ! requires scratch media. (Must not contain useable data).
; M  0219  1  !-
; M  0220  1
; M  0221  1  word ( (Status_Scratch_Media AND 1) ^ 0 )   ! Define the status word
;    0222  1     %,
;    0223  1
;    0224  1 !+
;    0225  1 !   This macro builds a DIAGNOSTIC block
;    0226  1 !-
;    0227  1
;    0228  1     INIT_DIAGBLK ( Diagnostic_Name,
;    0229  1      Set_Flags_Args,
;    0230  1        Set_Event_Args,
;    0231  1      Start_Qualifiers,
;    0232  1      Test_Start_Text,
; M  0233  1      RunTime ) =      ! [06]
; M  0234  1  word (%Length),
; M  0235  1  byte (%ascic %string(Diagnostic_Name)),
; M  0236  1  byte (%ascic %string(Set_Flags_Args)),
; M  0237  1  byte (%ascic %string(Set_Event_Args)),
; M  0238  1  byte (%ascic %string(Start_Qualifiers)),
; M  0239  1  byte (%ascic %string(Test_Start_Text)),
; M  0240  1  byte (%ascic %string(RunTime))
;    0241  1     %,
;    0242  1
;    0243  1 !+
;    0244  1 !   This macro just terminates the support block
;    0245  1 !-
;    0246  1
; M  0247  1     SUPBLK_END =
; M  0248  1  )
;    0249  1     %;
;    0250  1
;    0251  1
;    0252  1
```

```
;    0253  1 %SBTTL 'Support Data Base ID Block'
;    0254  1
; P  0255  1  SUPPORT_DBASE_ID
; P  0256  1
;    0257  1    (Version = ' V02.0');
```

```
;    0258  1 %SBTTL 'KA730 Offline Support Block'
;    0259  1
; P  0260  1  INIT_SUPBLK_HEAD
; P  0261  1    (
; P  0262  1    Device_Name = KA730,
; P  0263  1    Test_Category = MEN$K_Tstctg_CPU,
; P  0264  1    Test_Class = MEN$K_Tstcla_CPU,
; P  0265  1    Testcnfg_No = 1,
; P  0266  1    Diagnostic_Level = '3'
;    0267  1    ),
; P  0268  1  INIT_TSTCNFG_BLK
; P  0269  1    (
; P  0270  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0271  1    Diagblk_No = 5
;    0272  1    ),
; P  0273  1  INIT_DIAGBLK
; P  0274  1    (
; P  0275  1    Diagnostic_Name = ENKAX,
; P  0276  1    Set_Flags_Args = 'QUICK',
; P  0277  1    Set_Event_Args = '',
; P  0278  1    Start_Qualifiers = '',
; P  0279  1    Test_Start_Text = 'PART 1',
; P  0280  1    RunTime = '00:45'
;    0281  1    ),
; P  0282  1  INIT_DIAGBLK
; P  0283  1    (
; P  0284  1    Diagnostic_Name - EVKAB,
; P  0285  1    Set_Flags_Args = 'QUICK',
; P  0286  1    Set_Event_Args = '',
; P  0287  1    Start_Qualifiers = '',
; P  0288  1    Test_Start_Text = 'PART 2',
; P  0289  1    RunTime = '02:00'
;    0290  1    ),
; P  0291  1  INIT_DIAGBLK
; P  0292  1    (
; P  0293  1    Diagnostic_Name = EVKAC,
; P  0294  1    Set_Flags_Args = 'QUICK',
; P  0295  1    Set_Event_Args = '',
; P  0296  1    Start_Qualifiers = '',
; P  0297  1    Test_Start_Text = 'PART 3',
; P  0298  1    RunTime = '03:00'
;    0299  1    ),
; P  0300  1  INIT_DIAGBLK
; P  0301  1    (
; P  0302  1    Diagnostic_Name = EVKAD,
; P  0303  1    Set_Flags_Args = 'QUICK',
; P  0304  1    Set_Event_Args = '',
; P  0305  1    Start_Qualifiers - '',
; P  0306  1    Test_Start_Text = 'PART 4',
; P  0307  1    RunTime - '02:45'
;    0308  1    ),
; P  0309  1  INIT_DIAGBLK
; P  0310  1    (
; P  0311  1    Diagnostic_Name - EVKAE,
; P  0312  1    Set_Flags_Args - 'QUICK',
```

```
; P 0313  1     Set_Event_Args = '',
; P 0314  1     Start_Qualifiers = '',
; P 0315  1     Test_Start_Text = 'PART 5',
; P 0316  1     RunTime = '02:00'
;   0317  1     )
;   0318  1   SUPBLK_END;
;   0319  1
```

```
;    0320  1 %SBTTL 'KA750 Offline Support Block'
;    0321  1
; P  0322  1  INIT_SUPBLK_HEAD
; P  0323  1    (
; P  0324  1    Device_Name = KA750,
; P  0325  1    Test_Category = MEN$K_Tstctg_CPU,
; P  0326  1    Test_Class = MEN$K_Tstcla_CPU,
; P  0327  1    Testcnfg_No = 1,
; P  0328  1    Diagnostic_Level = '3'
;    0329  1    ),
; P  0330  1  INIT_TSTCNFG_BLK
; P  0331  1    (
; P  0332  1    DevTstPrep_Code = MEN$K_DevTstPrep Null,
; P  0333  1    Diagblk_No = 4
;    0334  1    ),
; P  0335  1  INIT_DIAGBLK
; P  0336  1    (
; P  0337  1    Diagnostic_Name = EVKAB,
; P  0338  1    Set_Flags_Args = 'QUICK',
; P  0339  1    Set_Event_Args = '',
; P  0340  1    Start_Qualifiers = '',
; P  0341  1    Test_Start_Text = 'PART 1',
; P  0342  1    RunTime = '02:00'
;    0343  1    ),
; P  0344  1  INIT_DIAGBLK
; P  0345  1    (
; P  0346  1    Diagnostic_Name = EVKAC,
; P  0347  1    Set_Flags_Args = 'QUICK',
; P  0348  1    Set_Event_Args = '',
; P  0349  1    Start_Qualifiers = '',
; P  0350  1    Test_Start_Text = 'PART 2',
; P  0351  1    RunTime = '03:00'
;    0352  1    ),
; P  0353  1  INIT_DIAGBLK
; P  0354  1    (
; P  0355  1    Diagnostic_Name = EVKAD,
; P  0356  1    Set_Flags_Args = 'QUICK',
; P  0357  1    Set_Event_Args = '',
; P  0358  1    Start_Qualifiers = '',
; P  0359  1    Test_Start_Text - 'PART 3',
; P  0360  1    RunTime = '02:45'
;    0361  1    ),
; P  0362  1  INIT_DIAGBLK
; P  0363  1    (
; P  0364  1    Diagnostic_Name = EVKAE,
; P  0365  1    Set_Flags_Args = 'QUICK',
; P  0366  1    Set_Event_Args = '',
; P  0367  1    Start_Qualifiers = '',
; P  0368  1    Test_Start_Text = 'PART 4',
; P  0369  1    RunTime = '02:00'
;    0370  1    )
;    0371  1  SUPBLK_END;
;    0372  1
```

```
;   0373  1 %SBTTL 'KA780 Offline Support Block'
;   0374  1
; P 0375  1  INIT_SUPBLK_HEAD
; P 0376  1    (
; P 0377  1    Device_Name = KA780,
; P 0378  1    Test_Category = MEN$K_Tstctg_CPU,
; P 0379  1    Test_Class = MEN$K_Tstcla_CPU,
; P 0380  1    Testcnfg_No = 1,
; P 0381  1    Diagnostic_Level = '3'
;   0382  1    ),
; P 0383  1  INIT_TSTCNFG_BLK
; P 0384  1    (
; P 0385  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 0386  1    Diagblk_No = 4
;   0387  1    ),
; P 0388  1  INIT_DIAGBLK
; P 0389  1    (
; P 0390  1    Diagnostic_Name = EVKAB,
; P 0391  1    Set_Flags_Args = 'QUICK',
; P 0392  1    Set_Event_Args = '',
; P 0393  1    Start_Qualifiers = '',
; P 0394  1    Test_Start_Text = 'PART 1',
; P 0395  1    RunTime = '02:00'
;   0396  1    ),
; P 0397  1  INIT_DIAGBLK
; P 0398  1    (
; P 0399  1    Diagnostic_Name = EVKAC,
; P 0400  1    Set_Flags_Args = 'QUICK',
; P 0401  1    Set_Event_Args = '',
; P 0402  1    Start_Qualifiers = '',
; P 0403  1    Test_Start_Text = 'PART 2',
; P 0404  1    RunTime = '03:00'
;   0405  1    ),
; P 0406  1  INIT_DIAGBLK
; P 0407  1    (
; P 0408  1    Diagnostic_Name = EVKAD,
; P 0409  1    Set_Flags_Args = 'QUICK',
; P 0410  1    Set_Event_Args = '',
; P 0411  1    Start_Qualifiers = '',
; P 0412  1    Test_Start_Text = 'PART 3',
; P 0413  1    RunTime = '02:45'
;   0414  1    ),
; P 0415  1  INIT_DIAGBLK
; P 0416  1    (
; P 0417  1    Diagnostic_Name = EVKAE,
; P 0418  1    Set_Flags_Args = 'QUICK',
; P 0419  1    Set_Event_Args = '',
; P 0420  1    Start_Qualifiers = '',
; P 0421  1    Test_Start_Text = 'PART 4',
; P 0422  1    RunTime = '02:00'
;   0423  1    )
;   0424  1  SUPBLK_END;
;   0425  1
```

```
;    0426  1 %SBTTL 'KA820 Offline Support Block'
;    0427  1
; P  0428  1  INIT_SUPBLK_HEAD
; P  0429  1    (
; P  0430  1    Device_Name = KA820,
; P  0431  1    Test_Category = MEN$K_Tstctg_CPU,
; P  0432  1    Test_Class = MEN$K_Tstcla_CPU,
; P  0433  1    Testcnfg_No = 1,
; P  0434  1    Diagnostic_Level = '3'
;    0435  1    ),
; P  0436  1  INIT_TSTCNFG_BLK
; P  0437  1    (
; P  0438  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0439  1    Diagblk_No = 3
;    0440  1    ),
; P  0441  1  INIT_DIAGBLK
; P  0442  1    (
; P  0443  1    Diagnostic_Name = EVKAB,
; P  0444  1    Set_Flags_Args = 'QUICK',
; P  0445  1    Set_Event_Args = '',
; P  0446  1    Start_Qualifiers = '',
; P  0447  1    Test_Start_Text = 'PART 1',
; P  0448  1    RunTime = '02:00'
;    0449  1    ),
; P  0450  1  INIT_DIAGBLK
; P  0451  1    (
; P  0452  1    Diagnostic_Name = EVKAC,
; P  0453  1    Set_Flags_Args = 'QUICK',
; P  0454  1    Set_Event_Args = '',
; P  0455  1    Start_Qualifiers = '',
; P  0456  1    Test_Start_Text = 'PART 2',
; P  0457  1    RunTime = '03:00'
;    0458  1    ),
; P  0459  1  INIT_DIAGBLK
; P  0460  1    (
; P  0461  1    Diagnostic_Name = EVKAE,
; P  0462  1    Set_Flags_Args = 'QUICK',
; P  0463  1    Set_Event_Args = '',
; P  0464  1    Start_Qualifiers = '',
; P  0465  1    Test_Start_Text = 'PART 3',
; P  0466  1    RunTime = '02:00'
;    0467  1    )
;    0468  1  SUPBLK_END;
;    0469  1
```

```
;    0470  : %SBTTL 'KA86 Offline Support Block'
;    0471  1
; P  0472  1  INIT_SUPBLK_HEAD
; P  0473  1    (
; P  0474  1    Device_Name = KA86,
; P  0475  1    Test_Category = MEN$K_Tstctg_CPU,
; P  0476  1    Test_Class = MEN$K_Tstcla_CPU,
; P  0477  1    Testcnfg_No = 1,
; P  0478  1    Diagnostic_Level = '3'
;    0479  1    ),
; P  0480  1  INIT_TSTCNFG_BLK
; P  0481  1    (
; P  0482  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0483  1    Diagblk_No = 4
;    0484  1    ),
; P  0485  1  INIT_DIAGBLK
; P  0486  1    (
; P  0487  1    Diagnostic Name = EVKAB,
; P  0488  1    Set_Flags_Args = 'QUICK',
; P  0489  1    Set_Event_Args = '',
; P  0490  1    Start_Qualifiers = '',
; P  0491  1    Test_Start_Text = 'PART 1',
; P  0492  1    RunTime = '02:00'
;    0493  1    ),
; P  0494  1  INIT_DIAGBLK
; P  0495  1    (
; P  0496  1    Diagnostic_Name = EVKAC,
; P  0497  1    Set_Flags_Args = 'QUICK',
; P  0498  1    Set_Event_Args = '',
; P  0499  1    Start_Qualifiers = '',
; P  0500  1    Test_Start_Text = 'PART 2',
; P  0501  1    RunTime = '03:00'
;    0502  1    ),
; P  0503  1  INIT_DIAGBLK
; P  0504  1    (
; P  0505  1    Diagnostic Name = EVKAD,
; P  0506  1    Set_Flags_Args = 'QUICK',
; P  0507  1    Set_Event_Args = '',
; P  0508  1    Start_Qualifiers = '',
; P  0509  1    Test_Start_Text = 'PART 3',
; P  0510  1    RunTime = '02:45'
;    0511  1    ),
; P  0512  1  INIT_DIAGBLK
; P  0513  1    (
; P  0514  1    Diagnostic_Name = EVKAE,
; P  0515  1    Set_Flags_Args = 'QUICK',
; P  0516  1    Set_Event_Args = '',
; P  0517  1    Start_Qualifiers = '',
; P  0518  1    Test_Start_Text = 'PART 4',
; P  0519  1    RunTime = '02:00'
;    0520  1    )
;    0521  1  SUPBLK_END;
;    0522  1
```

```
;    0523  1 %SBTTL 'TS11 Offline Support Block'
;    0524  1
; P  0525  1  INIT_SUPBLK_HEAD
; P  0526  1    (
; P  0527  1    Device_Name = TS11,
; P  0528  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  0529  1    Test_Class = MEN$K_Tstcla_Tape,
; P  0530  1    Testcnfg_No = 1,
; P  0531  1    Diagnostic_Level = '3'
;    0532  1    ),
; P  0533  1  INIT_TSTCNFG_BLK
; P  0534  1    (
; P  0535  1    DevTstPrep_Code = MEN$K_DevTstPrep_3,
; P  0536  1    Diagblk_No = 1,
; P  0537  1    Status_Scratch_Media = True
;    0538  1    ),
; P  0539  1  INIT_DIAGBLK
; P  0540  1    (
; P  0541  1    Diagnostic_Name = EVMAD,
; P  0542  1    Set_Flags_Args = 'QUICK',
; P  0543  1    Set_Event_Args = '',
; P  0544  1    Start_Qualifiers = '',
; P  0545  1    Test_Start_Text = '',
; P  0546  1    RunTime = '07:30'
;    0547  1    )
;    0548  1  SUPBLK_END;
;    0549  1
;    0550  1
;    0551  1
```

```
;    0552  1
;    0553  1 %SBTTL 'TU80 Offline Support Block'
;    0554  1
; P  0555  1  INIT_SUPBLK_HEAD
; P  0556  1    (
; P  0557  1    Device_Name = TU80,
; P  0558  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  0559  1    Test_Class = MEN$K_Tstcla_Tape,
; P  0560  1    Testcnfg_No = 1,
; P  0561  1    Diagnostic_Level = '3'
;    0562  1    ),
; P  0563  1  INIT_TSTCNFG_BLK
; P  0564  1    (
; P  0565  1    DevTstPrep_Code = MEN$K_DevTstPrep_3,
; P  0566  1    Diagblk_No = 2,
; P  0567  1    Status_Scratch_Media = True
;    0568  1    ),
; P  0569  1  INIT_DIAGBLK
; P  0570  1    (
; P  0571  1    Diagnostic_Name = EVMBD,
; P  0572  1    Set_Flags_Args = 'QUICK',
; P  0573  1    Set_Event_Args = '',
; P  0574  1    Start_Qualifiers = '/SEC:DEFAULT',
; P  0575  1    Test_Start_Text = 'PART 1',
; P  0576  1    RunTime = '03:30'
;    0577  1    ),
; P  0578  1  INIT_DIAGBLK
; P  0579  1    (
; P  0580  1    Diagnostic_Name = EVMBE,
; P  0581  1    Set_Flags_Args = 'QUICK',
; P  0582  1    Set_Event_Args = '',
; P  0583  1    Start_Qualifiers = '/SEC:CRD',
; P  0584  1    Test_Start_Text = 'PART 2',
; P  0585  1    RunTime = '03:30'
;    0586  1    )
;    0587  1  SUPBLK_END;
;    0588  1
;    0589  1
;    0590  1
```

```
;    0591  1
;    0592  1 %SBTTL 'TU81 Offline Support Block'
;    0593  1
; P  0594  1  INIT_SUPBLK_HEAD
; P  0595  1    (
; P  0596  1    Device_Name = TU81,
; P  0597  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  0598  1    Test_Class = MEN$K_Tstcla_Tape,
; P  0599  1    Testcnfg_No = 1,
; P  0600  1    Diagnostic_Level = '3'
;    0601  1    ),
; P  0602  1  INIT_TSTCNFG_BLK
; P  0603  1    (
; P  0604  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0605  1    Diagblk_No = 1
;    0606  1    ),
; P  0607  1  INIT_DIAGBLK
; P  0608  1    (
; P  0609  1    Diagnostic_Name = EVMBB,
; P  0610  1    Set_Flags_Args = 'QUICK',
; P  0611  1    Set_Event_Args = '',
; P  0612  1    Start_Qualifiers = '/SEC:DEFAULT',
; P  0613  1    Test_Start_Text = 'PART 1',
; P  0614  1    RunTime = '03:30'
;    0615  1    )
;    0616  1  SUPBLK_END;
;    0617  1
;    0618  1
;    0619  1
```

```
;    0620  1 %SBTTL 'RK611 Offline Support Block'
; P  0621  1  INIT_SUPBLK_HEAD
; P  0622  1    (
; P  0623  1    Device_Name = RK611,
; P  0624  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  0625  1    Test_Class = MEN$K_Tstcla_Disk,
; P  0626  1    Testcnfg_No = 1,
; P  0627  1    Diagnostic_Level = '3'
;    0628  1    ),
; P  0629  1  INIT_TSTCNFG_BLK
; P  0630  1    (
; P  0631  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0632  1    Diagblk_No = 5
;    0633  1    ),
; P  0634  1  INIT_DIAGBLK
; P  0635  1    (
; P  0636  1    Diagnostic_Name = EVREA,
; P  0637  1    Set_Flags_Args = 'QUICK',
; P  0638  1    Set_Event_Args = '',
; P  0639  1    Start_Qualifiers = '',
; P  0640  1    Test_Start_Text = 'PART 1',
; P  0641  1    RunTime = '00:30'
;    0642  1    ),
; P  0643  1  INIT_DIAGBLK
; P  0644  1    (
; P  0645  1    Diagnostic_Name = EVREB,
; P  0646  1    Set_Flags_Args = 'QUICK',
; P  0647  1    Set_Event_Args = '',
; P  0648  1    Start_Qualifiers = '',
; P  0649  1    Test_Start_Text = 'PART 2',
; P  0650  1    RunTime = '01:30'
;    0651  1    ),
; P  0652  1  INIT_DIAGBLK
; P  0653  1    (
; P  0654  1    Diagnostic Name = EVREC,
; P  0655  1    Set_Flags_Args = 'QUICK',
; P  0656  1    Set_Event_Args = '',
; P  0657  1    Start_Qualifiers = '',
; P  0658  1    Test_Start_Text - 'PART 3' ,
; P  0659  1    RunTime = '01:30'
;    0660  1    ),
; P  0661  1  INIT_DIAGBLK
; P  0662  1    (
; P  0663  1    Diagnostic_Name = EVRED,
; P  0664  1    Set_Flags Args = 'QUICK',
; P  0665  1    Set_Event_Args = '',
; P  0666  1    Start_Qualifiers = '',
; P  0667  1    Test_Start_Text = 'PART 4',
; P  0668  1    RunTime = '00:30'
;    0669  1    ),
; P  0670  1  INIT_DIAGBLK
; P  0671  1    (
; P  0672  1    Diagnostic_Name = EVREE,
; P  0673  1    Set_Flags_Args - 'QUICK',
; P  0674  1    Set_Event_Args - '',
```

```
; P 0675  1     Start_Qualifiers = '',
; P 0676  1     Test_Start_Text = 'PART 5',
; P 0677  1     RunTime = '01:45'
;   0678  1     )
;   0679  1   SUPBLK_END;
;   0680  1
;   0681  1
;   0682  1
;   0683  1
```

```
;    0684  1
;    0685  1 %SBTTL 'RK711 Offline Support Block'
; P  0686  1  INIT_SUPBLK_HEAD
; P  0687  1    (
; P  0688  1    Device_Name = RK711,
; P  0689  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  0690  1    Test_Class = MEN$K_Tstcla_Disk,
; P  0691  1    Testcnfg_No = 1,
; P  0692  1    Diagnostic_Level = '3'
;    0693  1    ),
; P  0694  1  INIT_TSTCNFG_BLK
; P  0695  1    (
; P  0696  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0697  1    Diagblk_No = 5
;    0698  1    ),
; P  0699  1  INIT_DIAGBLK
; P  0700  1    (
; P  0701  1    Diagnostic_Name = EVREA,
; P  0702  1    Set_Flags_Args = 'QUICK',
; P  0703  1    Set_Event_Args = '',
; P  0704  1    Start_Qualifiers = '',
; P  0705  1    Test_Start_Text = 'PART 1',
; P  0706  1    RunTime = '00:30'
;    0707  1    ),
; P  0708  1  INIT_DIAGBLK
; P  0709  1    (
; P  0710  1    Diagnostic_Name = EVREB,
; P  0711  1    Set_Flags_Args = 'QUICK',
; P  0712  1    Set_Event_Args = '',
; P  0713  1    Start_Qualifiers = '',
; P  0714  1    Test_Start_Text = 'PART 2',
; P  0715  1    RunTime = '01:30'
;    0716  1    ),
; P  0717  1  INIT_DIAGBLK
; P  0718  1    (
; P  0719  1    Diagnostic_Name = EVREC,
; P  0720  1    Set_Flags_Args = 'QUICK',
; P  0721  1    Set_Event_Args = '',
; P  0722  1    Start_Qualifiers = '',
; P  0723  1    Test_Start_Text = 'PART 3' ,
; P  0724  1    RunTime = '01:30'
;    0725  1    ),
; P  0726  1  INIT_DIAGBLK
; P  0727  1    (
; P  0728  1    Diagnostic_Name = EVRED,
; P  0729  1    Set_Flags_Args = 'QUICK',
; P  0730  1    Set_Event_Args = '',
; P  0731  1    Start_Qualifiers = '',
; P  0732  1    Test_Start_Text = 'PART 4',
; P  0733  1    RunTime = '00:30'
;    0734  1    ),
; P  0735  1  INIT_DIAGBLK
; P  0736  1    (
; P  0737  1    Diagnostic_Name = EVREE,
; P  0738  1    Set_Flags_Args = 'QUICK',
```

```
;P 0739  1     Set_Event_Args = '',
;P 0740  1     Start_Qualifiers = '',
;P 0741  1     Test_Start_Text = 'PART 5',
;P 0742  1     RunTime = '01:45'
;   0743  1     )
;   0744  1   SUPBLK_END;
;   0745  1
;   0746  1
;   0747  1
;   0748  1
```

```
;    0749  1 %SBTTL 'RK07 Offline Support Block'
; P  0750  1  INIT_SUPBLK_HEAD
; P  0751  1    (
; P  0752  1    Device_Name = RK07,
; P  0753  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  0754  1    Test_Class = MEN$K_Tstcla_Disk,
; P  0755  1    Testcnfg_No = 1,
; P  0756  1    Diagnostic_Level = '3'
;    0757  1    ),
; P  0758  1  INIT_TSTCNFG_BLK
; P  0759  1    (
; P  0760  1    DevTstPrep_Code = MEN$K_DevTstPrep_1,
; P  0761  1    Diagblk_No = 1
;    0762  1    ),
; P  0763  1  INIT_DIAGBLK
; P  0764  1    (
; P  0765  1    Diagnostic_Name = EVRAD,
; P  0766  1    Set_Flags_Args = 'QUICK',
; P  0767  1    Set_Event_Args = '',
; P  0768  1    Start_Qualifiers = '',
; P  0769  1    Test_Start_Text = '',
; P  0770  1    RunTime = '00:45'
;    0771  1    )
;    0772  1  SUPBLK END;
```

```
;    0773  1 %SBTTL 'RL02 Offline Support Block'
; P  0774  1  INIT_SUPBLK_HEAD
; P  0775  1    (
; P  0776  1    Device_Name = RL02,
; P  0777  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  0778  1    Test_Class - MEN$K_Tstcla_Disk,
; P  0779  1    Testcnfg_No = 1,
; P  0780  1    Diagnostic_Level = '3'
;    0781  1    ),
; P  0782  1  INIT_TSTCNFG_BLK
; P  0783  1    (
; P  0784  1    DevTstPrep_Code = MEN$K_DevTstPrep_1,
; P  0785  1    Diagblk_No = 1
;    0786  1    ),
; P  0787  1  INIT_DIAGBLK
; P  0788  1    (
; P  0789  1    Diagnostic_Name = EVRAD,
; P  0790  1    Set_Flags_Args = 'QUICK',
; P  0791  1    Set_Event_Args = '',
; P  0792  1    Start_Qualifiers = '',
; P  0793  1    Test_Start_Text = '',
; P  0794  1    RunTime = '00:45'
;    0795  1    )
;    0796  1  SUPBLK_END;
```

```
;    0797  1 %SBTTL 'R80 Offline Support Block'
; P  0798  1  INIT_SUPBLK_HEAD
; P  0799  1    (
; P  0800  1    Device_Name = R80,
; P  0801  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  0802  1    Test_Class = MEN$K_Tstcla_Disk,
; P  0803  1    Testcnfg_No = 1,
; P  0804  1    Diagnostic_Level = '3'
;    0805  1    ),
; P  0806  1  INIT_TSTCNFG_BLK
; P  0807  1    (
; P  0808  1    DevTstPrep_Code = MEN$K_DevTstPrep_4,
; P  0809  1    Diagblk_No = 1
;    0810  1    ),
; P  0811  1  INIT_DIAGBLK
; P  0812  1    (
; P  0813  1    Diagnostic_Name = EVRAD,
; P  0814  1    Set_Flags_Args = 'QUICK',
; P  0815  1    Set_Event_Args = '',
; P  0816  1    Start_Qualifiers = '',
; P  0817  1    Test_Start_Text = '',
; P  0818  1    RunTime = '01:00'
;    0819  1    )
;    0820  1  SUPBLK_END;
;    0821  1
;    0822  1
```

```
;    0823  1 %SBTTL 'RA60 Offline Support Block'
; P  0824  1  INIT_SUPBLK_HEAD
; P  0825  1    (
; P  0826  1    Device_Name = RA60,
; P  0827  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  0828  1    Test_Class = MEN$K_Tstcla_Disk,
; P  0829  1    Testcnfg_No = 1,
; P  0830  1    Diagnostic_Level = '3'
;    0831  1    ),
; P  0832  1  INIT_TSTCNFG_BLK
; P  0833  1    (
; P  0834  1    DevTstPrep_Code = MEN$K_DevTstPrep_1,
; P  0835  1    Diagblk_No = 1
;    0836  1    ),
; P  0837  1  INIT_DIAGBLK
; P  0838  1    (
; P  0839  1    Diagnostic_Name = EVRLA,
; P  0840  1    Set_Flags_Args = 'QUICK',
; P  0841  1    Set_Event_Args = '',
; P  0842  1    Start_Qualifiers = '/SECTION:CRD12',
; P  0843  1    Test_Start_Text = '',
; P  0844  1    RunTime = '01:00'
;    0845  1    )
;    0846  1  SUPBLK END;
;    0847  1
```

```
;    0848  1 %SBTTL 'RA80 Offline Support Block'
; P  0849  1  INIT_SUPBLK_HEAD
; P  0850  1    (
; P  0851  1    Device_Name = RA80,
; P  0852  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  0853  1    Test_Class = MEN$K_Tstcla_Disk,
; P  0854  1    Testcnfg_No = 1,
; P  0855  1    Diagnostic_Level = '3'
;    0856  1    ),
; P  0857  1  INIT_TSTCNFG_BLK
; P  0858  1    (
; P  0859  1    DevTstPrep_Code = MEN$K_DevTstPrep_4,
; P  0860  1    Diagblk_No = 1
;    0861  1    ),
; P  0862  1  INIT_DIAGBLK
; P  0863  1    (
; P  0864  1    Diagnostic_Name = EVRLA,
; P  0865  1    Set_Flags_Args = 'QUICK',
; P  0866  1    Set_Event_Args = '',
; P  0867  1    Start_Qualifiers = '/SECTION:CRD12',
; P  0868  1    Test_Start_Text = '',
; P  0869  1    RunTime = '02:45'
;    0870  1    )
;    0871  1  SUPBLK_END;
;    0872  1
```

```
;   0873  1 %SBTTL 'RA81 Offline Support Block'
; P 0874  1  INIT_SUPBLK_HEAD
; P 0875  1    (
; P 0876  1    Device_Name = RA81,
; P 0877  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 0878  1    Test_Class = MEN$K_Tstcla_Disk,
; P 0879  1    Testcnfg_No = 1,
; P 0880  1    Diagnostic_Level = '3'
;   0881  1    ),
; P 0882  1  INIT_TSTCNFG_BLK
; P 0883  1    (
; P 0884  1    DevTstPrep_Code = MEN$K_DevTstPrep_4,
; P 0885  1    Diagblk_No = 1
;   0886  1    ),
; P 0887  1  INIT_DIAGBLK
; P 0888  1    (
; P 0889  1    Diagnostic_Name = EVRLA,
; P 0890  1    Set_Flags_Args = 'QUICK',
; P 0891  1    Set_Event_Args = '',
; P 0892  1    Start_Qualifiers = '/SECTION:CRD12',
; P 0893  1    Test_Start_Text = '',
; P 0894  1    RunTime = '01:30'
;   0895  1    )
;   0896  1  SUPBLK_END;
;   0897  1
```

```
;    0898  1 %SBTTL 'DHU11 Offline Support Block'
; P  0899  1  INIT_SUPBLK_HEAD
; P  0900  1    (
; P  0901  1    Device_Name = DHU11,
; P  0902  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  0903  1    Test_Class = MEN$K_Tstcla_Comm,
; P  0904  1    Testcnfg_No = 1,
; P  0905  1    Diagnostic_Level = '3'
;    0906  1    ),
; P  0907  1  INIT_TSTCNFG_BLK
; P  0908  1    (
; P  0909  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0910  1    Diagblk_No = 1
;    0911  1    ),
; P  0912  1  INIT_DIAGBLK
; P  0913  1    (
; P  0914  1    Diagnostic_Name = EVDAI,
; P  0915  1    Set_Flags_Args = 'QUICK',
; P  0916  1    Set_Event_Args = '',
; P  0917  1    Start_Qualifiers = '',
; P  0918  1    Test_Start_Text = '',
; P  0919  1    RunTime = '00:30'
;    0920  1    )
;    0921  1  SUPBLK_END;
;    0922  1
;    0923  1
```

```
;   0924  1
```

```
;    0925  1 %SBTTL 'DMF32S Offline Support Block'
; P  0926  1  INIT_SUPBLK_HEAD
; P  0927  1    (
; P  0928  1    Device_Name = DMF32S,
; P  0929  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  0930  1    Test_Class = MEN$K_Tstcla_Comm,
; P  0931  1    Testcnfg_No = 1,
; P  0932  1    Diagnostic_Level = '3'
;    0933  1    ),
; P  0934  1  INIT_TSTCNFG_BLK
; P  0935  1    (
; P  0936  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0937  1    Diagb k_No = 1
;    0938  1    ),
; P  0939  1  INIT_DIAGBLK
; P  0940  1    (
; P  0941  1    Diagnostic_Name = EVDLB,
; P  0942  1    Set_Flags_Args = 'QUICK',
; P  0943  1    Set_Event_Args = '',
; P  0944  1    Start_Qualifiers = '',
; P  0945  1    Test_Start_Text = 'COMBO',
; P  0946  1    RunTime = '01:00'
;    0947  1    )
;    0948  1  SUPBLK END;
;    0949  1
;    0950  1
```

```
;    0951  1 %SBTTL 'DMF32A Offline Support Block'
; P  0952  1  INIT_SUPBLK_HEAD
; P  0953  1    (
; P  0954  1    Device_Name = DMF32A,
; P  0955  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  0956  1    Test_Class = MEN$K_Tstcla_Comm,
; P  0957  1    Testcnfg_No = 1,
; P  0958  1    Diagnostic_Level = '3'
;    0959  1    ),
; P  0960  1  INIT_TSTCNFG_BLK
; P  0961  1    (
; P  0962  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0963  1    Diagblk_No = 1
;    0964  1    ),
; P  0965  1  INIT_DIAGBLK
; P  0966  1    (
; P  0967  1    Diagnostic_Name = EVDLC,
; P  0968  1    Set_Flags_Args = 'QUICK',
; P  0969  1    Set_Event_Args = '',
; P  0970  1    Start_Qualifiers = '',
; P  0971  1    Test_Start_Text = 'COMBO',
; P  0972  1    RunTime = '00:30'
;    0973  1    )
;    0974  1  SUPBLK END;
```

```
;    0975  1 %SBTTL 'DMF32P Offline Support Block'
; P  0976  1  INIT_SUPBLK_HEAD
; P  0977  1    (
; P  0978  1    Device_Name = DMF32P,
; P  0979  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  0980  1    Test_Class = MEN$K_Tstcla_Comm,
; P  0981  1    Testcnfg_No = 1,
; P  0982  1    Diagnostic_Level = '3'
;    0983  1    ),
; P  0984  1  INIT_TSTCNFG_BLK
; P  0985  1    (
; P  0986  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  0987  1    Diagblk_No = 1
;    0988  1    ),
; P  0989  1  INIT_DIAGBLK
; P  0990  1    (
; P  0991  1    Diagnostic_Name = EVDLD,
; P  0992  1    Set_Flags_Args = 'QUICK',
; P  0993  1    Set_Event_Args = '',
; P  0994  1    Start_Qualifiers = '',
; P  0995  1    Test_Start_Text = 'COMBO',
; P  0996  1    RunTime = '00:30'
;    0997  1    )
;    0998  1  SUPBLK_END;
;    0999  1
;    1000  1
```

```
;    1001  1 %SBTTL 'DMR11 Offline Support Block'
; P  1002  1  INIT_SUPBLK_HEAD
; P  1003  1    (
; P  1004  1    Device_Name = DMR11,
; P  1005  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  1006  1    Test_Class = MEN$K_Tstcla_Comm,
; P  1007  1    Testcnfg_No = 1,
; P  1008  1    Diagnostic_Level = '3'
;    1009  1    ),
; P  1010  1  INIT_TSTCNFG_BLK
; P  1011  1    (
; P  1012  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  1013  1    Diagblk_No = 2
;    1014  1    ),
; P  1015  1  INIT_DIAGBLK
; P  1016  1    (
; P  1017  1    Diagnostic_Name = EVDXA,
; P  1018  1    Set_Flags_Args = 'QUICK',
; P  1019  1    Set_Event_Args = '',
; P  1020  1    Start_Qualifiers = '',
; P  1021  1    Test_Start_Text = 'PART 1',
; P  1022  1    RunTime = '00:30'
;    1023  1    ),
; P  1024  1  INIT_DIAGBLK
; P  1025  1    (
; P  1026  1    Diagnostic_Name = EVDMA,
; P  1027  1    Set_Flags_Args = 'QUICK',
; P  1028  1    Set_Event_Args = '',
; P  1029  1    Start_Qualifiers = '',
; P  1030  1    Test_Start_Text = 'PART 2',
; P  1031  1    RunTime = '00:30'
;    1032  1    )
;    1033  1  SUPBLK_END;
;    1034  1
;    1035  1
```

```
;    1036  1
;    1037  1 %SBTTL 'DEUNA Offline Support Block'
; P  1038  1  INIT_SUPBLK_HEAD
; P  1039  1    (
; P  1040  1    Device_Name = UNA11,
; P  1041  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  1042  1    Test_Class = MEN$K_Tstcla_Comm,
; P  1043  1    Testcnfg_No = 1,
; P  1044  1    Diagnostic_Level = '3'
;    1045  1    ),
; P  1046  1  INIT_TSTCNFG_BLK
; P  1047  1    (
; P  1048  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  1049  1    Diagblk_No = 1
;    1050  1    ),
; P  1051  1  INIT_DIAGBLK
; P  1052  1    (
; P  1053  1    Diagnostic_Name = EVDWA,
; P  1054  1    Set_Flags_Args = '',
; P  1055  1    Set_Event_Args = '',
; P  1056  1    Start_Qualifiers = '',
; P  1057  1    Test_Start_Text = '',
; P  1058  1    RunTime = '02:00'
;    1059  1    )
;    1060  1  SUPBLK_END;
;    1061  1
;    1062  1
```

```
;    1063  1 %SBTTL 'DR11W Offline Support Block'
; P  1064  1  INIT_SUPBLK_HEAD
; P  1065  1    (
; P  1066  1    Device_Name = DR11W,
; P  1067  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  1068  1    Test_Class = MEN$K_Tstcla_Real,
; P  1069  1    Testcnfg_No = 1,
; P  1070  1    Diagnostic_Level = '3'
;    1071  1    ),
; P  1072  1  INIT_TSTCNFG_BLK
; P  1073  1    (
; P  1074  1    DevTstPrep Code = MEN$K_DevTstPrep_Null,
; P  1075  1    Diagblk_No = 2
;    1076  1    ),
; P  1077  1  INIT_DIAGBLK
; P  1078  1    (
; P  1079  1    Diagnostic_Name = EVDRE,
; P  1080  1    Set_Flags_Args = 'QUICK',
; P  1081  1    Set_Event_Args = '',
; P  1082  1    Start_Qualifiers = '',
; P  1083  1    Test_Start_Text = 'PART 1',
; P  1084  1    RunTime = '00:15'
;    1085  1    ),
; P  1086  1  INIT_DIAGBLK
; P  1087  1    (
; P  1088  1    Diagnostic_Name = EVDRB,
; P  1089  1    Set_Flags_Args = 'QUICK',
; P  1090  1    Set_Event_Args = '',
; P  1091  1    Start_Qualifiers = '',
; P  1092  1    Test_Start_Text = 'PART 2',
; P  1093  1    RunTime = '00:15'
;    1094  1    )
;    1095  1  SUPBLK_END;
;    1096  1
;    1097  1
```

```
;    1098  1
;    1099  1 %SBTTL 'DMP11 Offline Support Block'
; P  1100  1  INIT_SUPBLK_HEAD
; P  1101  1    (
; P  1102  1    Device_Name = DMP11,
; P  1103  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  1104  1    Test_Class = MEN$K_Tstcla_COMM,
; P  1105  1    Testcnfg_No = 1,
; P  1106  1    Diagnostic_Level = '3'
;    1107  1    ),
; P  1108  1  INIT_TSTCNFG_BLK
; P  1109  1    (
; P  1110  1    DevTstPrep_Code = MEN$K DevTstPrep_Null,
; P  1111  1    Diagblk_No = 1
;    1112  1    ),
; P  1113  1  INIT_DIAGBLK
; P  1114  1    (
; P  1115  1    Diagnostic_Name = EVDMB,
; P  1116  1    Set_Flags_Args = '',
; P  1117  1    Set_Event_Args = '',
; P  1118  1    Start_Qualifiers = '',
; P  1119  1    Test_Start_Text = '',
; P  1120  1    RunTime = '01:00'
;    1121  1    )
;    1122  1  SUPBLK_END;
;    1123  1
;    1124  1
;    1125  1
;    1126  1
```

```
;    1127  1 %SBTTL 'DUP11 Offline Support Block'
; P  1128  1  INIT_SUPBLK_HEAD
; P  1129  1    (
; P  1130  1    Device_Name = DUP11,
; P  1131  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  1132  1    Test_Class = MEN$K_Tstcla_COMM,
; P  1133  1    Testcnfg_No = 1,
; P  1134  1    Diagnostic_Level = '3'
;    1135  1    ),
; P  1136  1  INIT_TSTCNFG_BLK
; P  1137  1    (
; P  1138  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  1139  1    Diagblk_No = 1
;    1140  1    ),
; P  1141  1  INIT_DIAGBLK
; P  1142  1    (
; P  1143  1    Diagnostic_Name = EVDUP,
; P  1144  1    Set_Flags_Args = 'QUICK',
; P  1145  1    Set_Event_Args = '',
; P  1146  1    Start_Qualifiers = '',
; P  1147  1    Test_Start_Text = '',
; P  1148  1    RunTime = '00:10'
;    1149  1    )
;    1150  1  SUPBLK_END;
;    1151  1
;    1152  1
;    1153  1
;    1154  1
;    1155  1
;    1156  1
```

```
;    1157  1 %SBTTL 'DZ11 Offline Support Block'
; P  1158  1  INIT_SUPBLK_HEAD
; P  1159  1    (
; P  1160  1    Device_Name = XDZ11,
; P  1161  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  1162  1    Test_Class = MEN$K_Tstcla_TERM,
; P  1163  1    Testcnfg_No = 1,
; P  1164  1    Diagnostic_Level = '3'
;    1165  1    ),
; P  1166  1  INIT_TSTCNFG_BLK
; P  1167  1    (
; P  1168  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  1169  1    Diagblk_No = 1
;    1170  1    ),
; P  1171  1  INIT_DIAGBLK
; P  1172  1    (
; P  1173  1    Diagnostic_Name = EVDAA,
; P  1174  1    Set_Flags_Args = 'QUICK',
; P  1175  1    Set_Event_Args = '',
; P  1176  1    Start_Qualifiers = '',
; P  1177  1    Test_Start_Text = '',
; P  1178  1    RunTime = '00:00'
;    1179  1    )
;    1180  1  SUPBLK_END;
;    1181  1
;    1182  1
;    1183  1
;    1184  1
;    1185  1
;    1186  1
```

```
;    1187  1
;    1188  1 %SBTTL 'DZ32 Offline Support Block'
; P  1189  1  INIT_SUPBLK_HEAD
; P  1190  1    (
; P  1191  1    Device_Name = DZ32,
; P  1192  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P  1193  1    Test_Class = MEN$K_Tstcla_TERM,
; P  1194  1    Testcnfg_No = 1,
; P  1195  1    Diagnostic_Level = '3'
;    1196  1    ),
; P  1197  1  INIT_TSTCNFG_BLK
; P  1198  1    (
; P  1199  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  1200  1    Diagblk_No = 1
;    1201  1    ),
; P  1202  1  INIT_DIAGBLK
; P  1203  1    (
; P  1204  1    Diagnostic_Name = EVDAB,
; P  1205  1    Set_Flags_Args = 'QUICK',
; P  1206  1    Set_Event_Args = '',
; P  1207  1    Start_Qualifiers = '',
; P  1208  1    Test_Start_Text = '',
; P  1209  1    RunTime = '06:00'
;    1210  1    )
;    1211  1  SUPBLK_END;
;    1212  1
;    1213  1
;    1214  1
;    1215  1
;    1216  1
```

```
;   1217  1
;   1218  1 %SBTTL 'RC25 Offline Support Block'
;   1219  1             ! [03]
; P 1220  1  INIT_SUPBLK_HEAD
; P 1221  1    (
; P 1222  1    Device_Name = RC25,
; P 1223  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1224  1    Test_Class = MEN$K_Tstcla_Disk,
; P 1225  1    Testcnfg_No = 1,
; P 1226  1    Diagnostic_Level = '3'
;   1227  1    ),
; P 1228  1  INIT_TSTCNFG_BLK
; P 1229  1    (
; P 1230  1    DevTstPrep_Code = MEN$K_DevTstPrep_6,
; P 1231  1    Diagblk_No = 2
;   1232  1    ),
; P 1233  1  INIT_DIAGBLK
; P 1234  1    (
; P 1235  1    Diagnostic_Name = EVRMA,
; P 1236  1    Set_Flags_Args = 'QUICK',
; P 1237  1    Set_Event_Args = '',
; P 1238  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1239  1    Test_Start_Text = '',
; P 1240  1    RunTime = '01:00'
;   1241  1    ),
; P 1242  1  INIT_DIAGBLK
; P 1243  1    (
; P 1244  1    Diagnostic_Name = EVRMB,
; P 1245  1    Set_Flags_Args = 'QUICK',
; P 1246  1    Set_Event_Args = '',
; P 1247  1    Start_Qualifiers = '/SECTION:CRD',
; P 1248  1    Test_Start_Text = '',
; P 1249  1    RunTime = '01:00'
;   1250  1    )
;   1251  1  SUPBLK_END;
;   1252  1
;   1253  1
;   1254  1
```

```
;   1255  1
;   1256  1 %SBTTL 'RCF25 Offline Support Block'
;   1257  1             ! [03]
; P 1258  1  INIT_SUPBLK_HEAD
; P 1259  1    (
; P 1260  1    Device_Name = RCF25,
; P 1261  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1262  1    Test_Class = MEN$K_Tstcla_Disk,
; P 1263  1    Testcnfg_No = 1,
; P 1264  1    Diagnostic_Level = '3'
;   1265  1    ),
; P 1266  1  INIT_TSTCNFG_BLK
; P 1267  1    (
; P 1268  1    DevTstPrep_Code = MEN$K_DevTstPrep_6,
; P 1269  1    Diagblk_No = 2
;   1270  1    ),
; P 1271  1  INIT_DIAGBLK
; P 1272  1    (
; P 1273  1    Diagnostic_Name = EVRMA,
; P 1274  1    Set_Flags_Args = 'QUICK',
; P 1275  1    Set_Event_Args = '',
; P 1276  1    Start_Qualifiers = '/SECTION:CRD',
; P 1277  1    Test_Start_Text = '',
; P 1278  1    RunTime = '01:00'
;   1279  1    ),
; P 1280  1  INIT_DIAGBLK
; P 1281  1    (
; P 1282  1    Diagnostic_Name = EVRMB,
; P 1283  1    Set_Flags_Args = 'QUICK',
; P 1284  1    Set_Event_Args = '',
; P 1285  1    Start_Qualifiers = '/SECTION:CRD',
; P 1286  1    Test_Start_Text = '',
; P 1287  1    RunTime = '01:00'
;   1288  1    )
;   1289  1  SUPBLK_END;
;   1290  1
;   1291  1
```

```
;   1292  1
;   1293  1 %SBTTL 'RX31 Offline Support block'
;   1294  1
; P 1295  1  INIT_SUPBLK_HEAD
; P 1296  1  (
; P 1297  1  Device_Name = RX31,
; P 1298  1  Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1299  1  Test_Class = MEN$K_Tstcla_Disk,
; P 1300  1  Testcnfg_No=1,
; P 1301  1    Diagnostic_Level = '3'
;   1302  1  ),
; P 1303  1  INIT_TSTCNFG_BLK
; P 1304  1  (
; P 1305  1    DevTstPrep_Code = MEN$K DevTstPrep_7,
; P 1306  1    Diagblk_No = 1,
; P 1307  1    Status_Scratch_Media = False
;   1308  1    ),
; P 1309  1  INIT_DIAGBLK
; P 1310  1    (
; P 1311  1    Diagnostic_Name = EVRNB,
; P 1312  1    Set_Flags_Args    = 'QUICK',
; P 1313  1    Set_Event_Args    = '',
; P 1314  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1315  1    Test_Start_Text   = '',
; P 1316  1    RunTime           = '04:00'
;   1317  1    )
;   1318  1  SUPBLK_END;
```

```
;   1319  1 %SBTTL 'RX02 Offline Support block'
;   1320  1
; P 1321  1  INIT_SUPBLK_HEAD
; P 1322  1  (
; P 1323  1  Device_Name = RX02,
; P 1324  1  Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1325  1  Test_Class = MEN$K_Tstcla_Disk,
; P 1326  1  Testcnfg_No=1,
; P 1327  1    Diagnostic_Level = '3'
;   1328  1  ),
; P 1329  1  INIT_TSTCNFG_BLK
; P 1330  1  (
; P 1331  1    DevTstPrep_Code = MEN$K DevTstPrep 7,
; P 1332  1    Diagblk_No = 1,
; P 1333  1    Status_Scratch_Media = False
;   1334  1    ),
; P 1335  1  INIT_DIAGBLK
; P 1336  1    (
; P 1337  1    Diagnostic_Name = EVRIA,
; P 1338  1    Set_Flags_Args    = 'QUICK',
; P 1339  1    Set_Event_Args    = '',
; P 1340  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1341  1    Test_Start_Text   = '',
; P 1342  1    RunTime           = '00:00'
;   1343  1    )
;   1344  1  SUPBLK_END;
```

```
;   1345  1 %SBTTL 'RX32 Offline Support block'
;   1346  1
; P 1347  1  INIT_SUPBLK_HEAD
; P 1348  1  (
; P 1349  1  Device_Name = RX32,
; P 1350  1  Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1351  1  Test_Class = MEN$K_Tstcla_Disk,
; P 1352  1  Testcnfg_No=1,
; P 1353  1  Diagnostic_Level = '3'
;   1354  1  ),
; P 1355  1  INIT_TSTCNFG_BLK
; P 1356  1  (
; P 1357  1    DevTstPrep_Code = MEN$K_DevTstPrep 7,
; P 1358  1    Diagblk_No = 1,
; P 1359  1    Status_Scratch_Media = False
;   1360  1    ),
; P 1361  1  INIT_DIAGBLK
; P 1362  1    (
; P 1363  1    Diagnostic_Name = EVRNB,
; P 1364  1    Set_Flags_Args    = 'QUICK',
; P 1365  1    Set_Event_Args    = '',
; P 1366  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1367  1    Test_Start_Text   = '',
; P 1368  1    RunTime           = '04:00'
;   1369  1    )
;   1370  1  SUPBLK_END;
```

```
;    1371  1 %SBTTL 'RX50 Offline Support block'
;    1372  1
; P  1373  1  INIT_SUPBLK_HEAD
; P  1374  1  (
; P  1375  1  Device_Name = RX50,
; P  1376  1  Test_Category = MEN$K_Tstctg_IO_Unit,
; P  1377  1  Test_Class = MEN$K_Tstcla_Disk,
; P  1378  1  Testcnfg_No=1,
; P  1379  1  Diagnostic_Level = '3'
;    1380  1  ),
; P  1381  1  INIT_TSTCNFG_BLK
; P  1382  1  (
; P  1383  1    DevTstPrep_Code = MEN$K_DevTstPrep_7,
; P  1384  1    Diagblk_No = 1,
; P  1385  1    Status_Scratch_Media = False
;    1386  1    ),
; P  1387  1  INIT_DIAGBLK
; P  1388  1    (
; P  1389  1    Diagnostic_Name = EVRNB,
; P  1390  1    Set_Flags_Args   = 'QUICK',
; P  1391  1    Set_Event_Args   = '',
; P  1392  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  1393  1    Test_Start_Text  = '',
; P  1394  1    RunTime          = '04:00'
;    1395  1    )
;    1396  1  SUPBLK_END;
;    1397  1
;    1398  1 %SBTTL 'RX180 Offline Support block'
;    1399  1
; P  1400  1  INIT_SUPBLK_HEAD
; P  1401  1  (
; P  1402  1  Device_Name = RX180,
; P  1403  1  Test_Category = MEN$K_Tstctg_IO_Unit,
; P  1404  1  Test_Class = MEN$K_Tstcla_Disk,
; P  1405  1  Testcnfg_No=1,
; P  1406  1  Diagnostic_Level = '3'
;    1407  1  ),
; P  1408  1  INIT_TSTCNFG_BLK
; P  1409  1  (
; P  1410  1    DevTstPrep_Code = MEN$K DevTstPrep_7,
; P  1411  1    Diagblk_No - 1,
; P  1412  1    Status_Scratch_Media = Fa se
;    1413  1    ),
; P  1414  1  INIT_DIAGBLK
; P  1415  1    (
; P  1416  1    Diagnostic Name = EVRNB,
; P  1417  1    Set_Flags Args   = 'QUICK',
; P  1418  1    Set_Event_Args   = '',
; P  1419  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  1420  1    Test_Start_Text  = '',
; P  1421  1    RunTime          = '04:00'
;    1422  1    )
;    1423  1  SUPBLK_END;
;    1424  1
```

```
;   1425  1
;   1426  1 %SBTTL 'TS05 Offline Support Block'
;   1427  1             ! [03]
; P 1428  1  INIT_SUPBLK_HEAD
; P 1429  1    (
; P 1430  1    Device_Name = TS05,
; P 1431  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1432  1    Test_Class = MEN$K_Tstcla_Tape,
; P 1433  1    Testcnfg_No = 1,
; P 1434  1    Diagnostic_Level = '3'
;   1435  1    ),
; P 1436  1  INIT_TSTCNFG_BLK
; P 1437  1    (
; P 1438  1    DevTstPrep_Code = MEN$K_DevTstPrep_3,
; P 1439  1    Diagblk_No = 1,
; P 1440  1    Status_Scratch_Media = True
;   1441  1    ),
; P 1442  1  INIT_DIAGBLK
; P 1443  1    (
; P 1444  1    Diagnostic_Name = EVMAI,
; P 1445  1    Set_Flags_Args = 'QUICK',
; P 1446  1    Set_Event_Args = '',
; P 1447  1    Start_Qualifiers = '/SECTION:CRD',
; P 1448  1    Test_Start_Text = '',
; P 1449  1    RunTime = '03:45'
;   1450  1    )
;   1451  1  SUPBLK_END;
;   1452  1
;   1453  1
```

```
;    1454  1 %SBTTL 'VS100 Offline Support block'
;    1455  1
; P  1456  1  INIT_SUPBLK_HEAD
; P  1457  1    (
; P  1458  1    Device_Name = VS100,
; P  1459  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  1460  1    Test_Class = MEN$K_Tstcla_Term,
; P  1461  1    Testcnfg_No = 1,
; P  1462  1    Diagnostic_Level = '3'
;    1463  1    ),
; P  1464  1  INIT_TSTCNFG_BLK
; P  1465  1    (
; P  1466  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  1467  1    Diagblk_No = 1,
; P  1468  1    Status_Scratch_Media = False
;    1469  1    ),
; P  1470  1  INIT_DIAGBLK
; P  1471  1    (
; P  1472  1    Diagnostic_Name = EVWAB,
; P  1473  1    Set_Flags_Args = '',
; P  1474  1    Set_Event_Args = '',
; P  1475  1    Start_Qualifiers = '',
; P  1476  1    Test_Start_Text = '',
; P  1477  1    RunTime = '08:00'
;    1478  1    )
;    1479  1  SUPBLK_END;
;    1480  1
```

```
;    1481  1 %SBTTL 'BCI750 Offline Support block'
;    1482  1
; P  1483  1  INIT_SUPBLK_HEAD
; P  1484  1    (
; P  1485  1    Device_Name = BCI750,
; P  1486  1    Test_Category = MEN$K_Tstctg_IO_Adaptor,
; P  1487  1    Test_Class = MEN$K_Tstcla_IO_Adaptor,
; P  1488  1    Testcnfg_No = 1,
; P  1489  1    Diagnostic_Level = '3'
;    1490  1    ),
; P  1491  1  INIT_TSTCNFG_BLK
; P  1492  1    (
; P  1493  1    DevTstPrep_Code = MEN$K DevTstPrep_Null,
; P  1494  1    Diagblk_No = 6,
; P  1495  1    Status_Scratch_Med'a = False
;    1496  1    ),
; P  1497  1  INIT_DIAGBLK
; P  1498  1    (
; P  1499  1    Diagnostic_Name = EVCKA,
; P  1500  1    Set_Flags_Args = '',
; P  1501  1    Set_Event_Args = '',
; P  1502  1    Start_Qualifiers = '',
; P  1503  1    Test_Start_Text = '',
; P  1504  1    RunTime = '00:22'
;    1505  1    ),
; P  1506  1  INIT_DIAGBLK
; P  1507  1    (
; P  1508  1    Diagnostic_Name = EVCKB,
; P  1509  1    Set_Flags_Args - '',
; P  1510  1    Set_Event_Args - '',
; P  1511  1    Start_Qualifiers - '',
; P  1512  1    Test_Start_Text = '',
; P  1513  1    RunTime = '07:00'
;    1514  1    ),
; P  1515  1  INIT_DIAGBLK
; P  1516  1    (
; P  1517  1    Diagnostic_Name - EVCKC,
; P  1518  1    Set_Flags_Args = '',
; P  1519  1    Set_Event_Args = '',
; P  1520  1    Start_Qualifiers = '',
; P  1521  1    Test_Start_Text = '',
; P  1522  1    RunTime = '10:00'
;    1523  1    ),
; P  1524  1  INIT_DIAGBLK
; P  1525  1    (
; P  1526  1    Diagnostic_Name = EVCKD,
; P  1527  1    Set_Flags_Args = '',
; P  1528  1    Set_Event_Args = '',
; P  1529  1    Start_Qualifiers = '',
; P  1530  1    Test_Start_Text = '',
; P  1531  1    RunTime = '05:00'
;    1532  1    ),
; P  1533  1  INIT_DIAGBLK
; P  1534  1    (
; P  1535  1    Diagnostic_Name = EVCKE,
```

```
; P 1536  1     Set_Flags_Args = '',
; P 1537  1     Set_Event_Args = '',
; P 1538  1     Start_Qualifiers = '',
; P 1539  1     Test_Start_Text = '',
; P 1540  1     RunTime = '05:00'
;   1541  1     ),
; P 1542  1   INIT_DIAGBLK
; P 1543  1     (
; P 1544  1     Diagnostic_Name = EVCKF,
; P 1545  1     Set_Flags_Args = '',
; P 1546  1     Set_Event_Args = '',
; P 1547  1     Start_Qualifiers = '',
; P 1548  1     Test_Start_Text = '',
; P 1549  1     RunTime = '04:00'
;   1550  1     )
;   1551  1   SUPBLK_END;
;   1552  1
```

```
;    1553  1 %SBTTL 'DWBUA Offline Support block'
;    1554  1
; P  1555  1  INIT_SUPBLK_HEAD
; P  1556  1    (
; P  1557  1    Device_Name = DWBUA,
; P  1558  1    Test_Category = MEN$K_Tstctg_IO_Adaptor,
; P  1559  1    Test_Class = MEN$K_Tstcla_IO_Adaptor,
; P  1560  1    Testcnfg_No = 1,
; P  1561  1    Diagnostic_Level = '3'
;    1562  1    ),
; P  1563  1  INIT_TSTCNFG_BLK
; P  1564  1    (
; P  1565  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  1566  1    Diagblk_No = 1,
; P  1567  1    Status_Scratch_Media = False
;    1568  1    ),
; P  1569  1  INIT_DIAGBLK
; P  1570  1    (
; P  1571  1    Diagnostic_Name = EVCBB,
; P  1572  1    Set_Flags_Args = '',
; P  1573  1    Set_Event_Args = '',
; P  1574  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  1575  1    Test_Start_Text = '',
; P  1576  1    RunTime = '00:22'
;    1577  1    )
;    1578  1  SUPBLK_END;
;    1579  1
```

```
;    1580  1
;    1581  1 %SBTTL 'VS125 Offline Support Block'
;    1582  1             ! [03]
;    1583  1 ! INIT_SUPBLK_HEAD
;    1584  1 !   (
;    1585  1 !   Device_Name = VS125,
;    1586  1 !   Test_Category = MEN$K_Tstctg_IO_Unit,
;    1587  1 !   Test_Class = MEN$K_Tstcla_Term,
;    1588  1 !   Testcnfg_No = 1,
;    1589  1 !   Diagnostic_Level = '3'
;    1590  1 !   ),
;    1591  1 ! INIT_TSTCNFG_BLK
;    1592  1 !   (
;    1593  1 !   DevTstPrep_Code = MEN$K_DevTstPrep_Null,
;    1594  1 !   Diagblk_No = 1,
;    1595  1 !   Status_Scratch_Media = False
;    1596  1 !   ),
;    1597  1 ! INIT_DIAGBLK
;    1598  1 !   (
;    1599  1 !   Diagnostic_Name = ENWCA,
;    1600  1 !   Set_Flags_Args = 'QUICK',
;    1601  1 !   Set_Event_Args = '',
;    1602  1 !   Start_Qualifiers = '/SECTION:DEFAULT',
;    1603  1 !   Test_Start_Text = '',
;    1604  1 !   RunTime = '03:00'
;    1605  1 !   )
;    1606  1 ! SUPBLK_END;
;    1607  1
```

```
;   1608  1 !**************************************************************************
;   1609  1 !+
;   1610  1 ! The items that follow are online diagnostics/devices. The Diagnostic
;   1611  1 ! level entry may contain 2 or 2R. The answer only indicates that
;   1612  1 ! the diagnostics following are online diagnostics not offline.
;   1613  1 !-
;   1614  1 !**************************************************************************
;   1615  1
;   1616  1 %SBTTL 'DMF32 Online Support Block'
;   1617  1
; P 1618  1  INIT_SUPBLK_HEAD
; P 1619  1    (
; P 1620  1    Device_Name = DMF32S,
; P 1621  1    Test_Category - MEN$K_Tstctg_IO_Controller,
; P 1622  1    Test_Class = MEN$K_Tstcla_Comm,
; P 1623  1    Testcnfg_No = 1,
; P 1624  1    Diagnostic_Level = '2R'
;   1625  1    ),
; P 1626  1  INIT_TSTCNFG_BLK
; P 1627  1    (
; P 1628  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 1629  1    Diagblk_No = 1
;   1630  1    ),
; P 1631  1  INIT_DIAGBLK
; P 1632  1    (
; P 1633  1    Diagnostic_Name = 'EVDLA',
; P 1634  1    Set_Flags_Args = 'QUICK',
; P 1635  1    Set_Event_Args = '',
; P 1636  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1637  1    Test_Start_Text = 'DMF32S',
; P 1638  1    RunTime = '00:00'
;   1639  1    )
;   1640  1  SUPBLK_END;
;   1641  1
;   1642  1
;   1643  1 %SBTTL 'DR11W Online Support Block'
;   1644  1
; P 1645  1  INIT_SUPBLK_HEAD
; P 1646  1    (
; P 1647  1    Device Name = DR11W,
; P 1648  1    Test_Category = MEN$K_Tstctg IO_Controller,
; P 1649  1    Test_Class = MEN$K_Tstcla_Real,
; P 1650  1    Testcnfg_No = 1,
; P 1651  1    Diagnostic_Level = '2'
;   1652  1    ),
; P 1653  1  INIT_TSTCNFG BLK
; P 1654  1    (
; P 1655  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 1656  1    Diagblk_No = 1
;   1657  1    ),
; P 1658  1  INIT_DIAGBLK
; P 1659  1    (
; P 1660  1    Diagnostic_Name = 'EVDRB',
; P 1661  1    Set_Flags_Args = 'QUICK',
; P 1662  1    Set_Event_Args = '',
```

```
;P 1663  1     Start_Qualifiers = '/SECTION:DEFAULT',
;P 1664  1     Test_Start_Text = 'DR11W',
;P 1665  1     RunTime = '00:00'
;   1666  1     )
;   1667  1   SUPBLK_END;
;   1668  1
;   1669  1
;   1670  1  %SBTTL 'KA730 Online Support Block'
;   1671  1
;P 1672  1   INIT_SUPBLK_HEAD
;P 1673  1     (
;P 1674  1     Device_Name = KA730,
;P 1675  1     Test_Category = MEN$K_Tstctg_CPU,
;P 1676  1     Test_Class = MEN$K_Tstcla_CPU,
;P 1677  1     Testcnfg_No = 1,
;P 1678  1     Diagnostic_Level = '2R'
;   1679  1     ),
;P 1680  1   INIT_TSTCNFG_BLK
;P 1681  1     (
;P 1682  1     DevTstPrep_Code = MEN$K_DevTstPrep_Null,
;P 1683  1     Diagblk_No = 3
;   1684  1     ),
;P 1685  1   INIT_DIAGBLK
;P 1686  1     (
;P 1687  1     Diagnostic_Name = EVKAB,
;P 1688  1     Set_Flags_Args = 'QUICK',
;P 1689  1     Set_Event_Args = '',
;P 1690  1     Start_Qualifiers = '/SECTION:DEFAULT',
;P 1691  1     Test_Start_Text = 'PART I',
;P 1692  1     RunTime = '00:45'
;   1693  1     ),
;P 1694  1   INIT_DIAGBLK
;P 1695  1     (
;P 1696  1     Diagnostic_Name = EVKAC,
;P 1697  1     Set_Flags_Args = 'QUICK',
;P 1698  1     Set_Event_Args = '',
;P 1699  1     Start_Qualifiers = '/SECTION:DEFAULT',
;P 1700  1     Test_Start_Text = 'PART II',
;P 1701  1     RunTime = '01:00'
;   1702  1     ),
;P 1703  1   INIT_DIAGBLK
;P 1704  1     (
;P 1705  1     Diagnostic_Name = EVKAD,
;P 1706  1     Set_Flags_Args = 'QUICK',
;P 1707  1     Set_Event_Args = '',
;P 1708  1     Start_Qualifiers = '/SECTION:DEFAULT',
;P 1709  1     Test_Start_Text = 'PART III',
;P 1710  1     RunTime = '02:00'
;   1711  1     )
;   1712  1   SUPBLK_END;
```

```
;   1713  1 %SBTTL 'KA750 Online Support Block'
;   1714  1
; P 1715  1  INIT_SUPBLK_HEAD
; P 1716  1    (
; P 1717  1    Device_Name = KA750,
; P 1718  1    Test_Category = MEN$K_Tstctg_CPU,
; P 1719  1    Test_Class = MEN$K_Tstcla_CPU,
; P 1720  1    Testcnfg_No = 1,
; P 1721  1    Diagnostic_Level = '2R'
;   1722  1    ),
; P 1723  1  INIT_TSTCNFG_BLK
; P 1724  1    (
; P 1725  1    DevTstPrep_Code = MEN$K_DevTstPrep Null,
; P 1726  1    Diagblk_No = 3
;   1727  1    ),
; P 1728  1  INIT_DIAGBLK
; P 1729  1    (
; P 1730  1    Diagnostic_Name = EVKAB,
; P 1731  1    Set_Flags_Args = 'QUICK',
; P 1732  1    Set_Event_Args = '',
; P 1733  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1734  1    Test_Start_Text = 'Part I',
; P 1735  1    RunTime = '00:45'
;   1736  1    ),
; P 1737  1  INIT_DIAGBLK
; P 1738  1    (
; P 1739  1    Diagnostic_Name = EVKAC,
; P 1740  1    Set_Flags_Args = 'QUICK',
; P 1741  1    Set_Event_Args = '',
; P 1742  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1743  1    Test_Start_Text = 'Part II',
; P 1744  1    RunTime = '01:00'
;   1745  1    ),
; P 1746  1  INIT_DIAGBLK
; P 1747  1    (
; P 1748  1    Diagnostic_Name = EVKAD,
; P 1749  1    Set_Flags_Args = 'QUICK',
; P 1750  1    Set_Event_Args = '',
; P 1751  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1752  1    Test_Start_Text = 'Part III',
; P 1753  1    RunTime = '02:00'
;   1754  1    )
;   1755  1  SUPBLK END;
```

```
;    1756  1 %SBTTL 'KA780 Online Support Block'
;    1757  1
; P  1758  1  INIT_SUPBLK_HEAD
; P  1759  1    (
; P  1760  1    Device_Name = KA780,
; P  1761  1    Test_Category = MEN$K_Tstctg_CPU,
; P  1762  1    Test_Class = MEN$K_Tstcla_CPU,
; P  1763  1    Testcnfg_No = 1,
; P  1764  1    Diagnostic_Level = '2R'
;    1765  1    ),
; P  1766  1  INIT_TSTCNFG_BLK
; P  1767  1    (
; P  1768  1    DevTstPrep_Code = MEN$K_DevTstPrep Null,
; P  1769  1    Diagblk_No = 3
;    1770  1    ),
; P  1771  1  INIT_DIAGBLK
; P  1772  1    (
; P  1773  1    Diagnostic_Name = EVKAB,
; P  1774  1    Set_Flags_Args = 'QUICK',
; P  1775  1    Set_Event_Args = '',
; P  1776  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  1777  1    Test_Start_Text = 'Part I',
; P  1778  1    RunTime = '00:45'
;    1779  1    ),
; P  1780  1  INIT_DIAGBLK
; P  1781  1    (
; P  1782  1    Diagnostic_Name = EVKAC,
; P  1783  1    Set_Flags_Args = 'QUICK',
; P  1784  1    Set_Event_Args = '',
; P  1785  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  1786  1    Test_Start_Text = 'Part II',
; P  1787  1    RunTime = '01:00'
;    1788  1    ),
; P  1789  1  INIT_DIAGBLK
; P  1790  1    (
; P  1791  1    Diagnostic_Name = EVKAD,
; P  1792  1    Set_Flags_Args = 'QUICK',
; P  1793  1    Set_Event_Args = '',
; P  1794  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  1795  1    Test_Start_Text = 'Part III',
; P  1796  1    RunTime = '02:00'
;    1797  1    )
;    1798  1  SUPBLK_END;
;    1799  1
;    1800  1 %SBTTL 'KA820 Online Support Block'
;    1801  1
; P  1802  1  INIT_SUPBLK_HEAD
; P  1803  1    (
; P  1804  1    Device_Name = KA820,
; P  1805  1    Test_Category = MEN$K_Tstctg_CPU,
; P  1806  1    Test_Class = MEN$K_Tstcla_CPU,
; P  1807  1    Testcnfg_No = 1,
; P  1808  1    Diagnostic_Level = '2R'
;    1809  1    ),
; P  1810  1  INIT_TSTCNFG_BLK
```

```
;P 1811  1     (
;P 1812  1     DevTstPrep_Code = MEN$K_DevTstPrep_Null,
;P 1813  1     Diagblk_No = 2
;   1814  1     ),
;P 1815  1   INIT_DIAGBLK
;P 1816  1     (
;P 1817  1     Diagnostic_Name = EVKAB,
;P 1818  1     Set_Flags_Args = 'QUICK',
;P 1819  1     Set_Event_Args = '',
;P 1820  1     Start_Qualifiers = '/SECTION:DEFAULT',
;P 1821  1     Test_Start_Text = 'Part I',
;P 1822  1     RunTime = '00:45'
;   1823  1     ),
;P 1824  1   INIT_DIAGBLK
;P 1825  1     (
;P 1826  1     Diagnostic_Name = EVKAC,
;P 1827  1     Set_Flags_Args = 'QUICK',
;P 1828  1     Set_Event_Args = '',
;P 1829  1     Start_Qualifiers = '/SECTION:DEFAULT',
;P 1830  1     Test_Start_Text = 'Part II',
;P 1831  1     RunTime = '01:00'
;   1832  1     )
;   1833  1   SUPBLK_END;
```

```
;   1834  1 %SBTTL 'KA86 Online Support Block'
;   1835  1
; P 1836  1  INIT_SUPBLK_HEAD
; P 1837  1    (
; P 1838  1    Device_Name = KA86,
; P 1839  1    Test_Category = MEN$K_Tstctg_CPU,
; P 1840  1    Test_Class = MEN$K_Tstcla_CPU,
; P 1841  1    Testcnfg_No = 1,
; P 1842  1    Diagnostic_Level = '2R'
;   1843  1    ),
; P 1844  1  INIT_TSTCNFG_BLK
; P 1845  1    (
; P 1846  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 1847  1    Diagblk_No = 3
;   1848  1    ),
; P 1849  1  INIT_DIAGBLK
; P 1850  1    (
; P 1851  1    Diagnostic_Name = EVKAB,
; P 1852  1    Set_Flags_Args = 'QUICK',
; P 1853  1    Set_Event_Args = '',
; P 1854  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1855  1    Test_Start_Text = 'Part I',
;   1856  1    RunTime = '00:45'   ),
; P 1857  1  INIT_DIAGBLK
; P 1858  1    (
; P 1859  1    Diagnostic_Name = EVKAC,
; P 1860  1    Set_Flags_Args = 'QUICK',
; P 1861  1    Set_Event_Args = '',
; P 1862  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1863  1    Test_Start_Text = 'Part II',
; P 1864  1    RunTime = '01:00'
;   1865  1    ),
; P 1866  1  INIT_DIAGBLK
; P 1867  1    (
; P 1868  1    Diagnostic_Name = EVKAD,
; P 1869  1    Set_Flags_Args = 'QUICK',
; P 1870  1    Set_Event_Args = '',
; P 1871  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1872  1    Test_Start_Text = 'Part III',
; P 1873  1    RunTime = '02:00'
;   1874  1    )
;   1875  1  SUPBLK_END;
```

```
;   1876  1 %SBTTL 'RC25 Online Support Block'
;   1877  1
; P 1878  1  INIT_SUPBLK_HEAD
; P 1879  1    (
; P 1880  1    Device_Name = RC25,
; P 1881  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1882  1    Test_Class = MEN$K_Tstcla_Disk,
; P 1883  1    Testcnfg_No = 1,
; P 1884  1    Diagnostic_Level = '2R'
;   1885  1    ),
; P 1886  1  INIT_TSTCNFG_BLK
; P 1887  1    (
; P 1888  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 1889  1    Diagblk_No = 1
;   1890  1    ),
; P 1891  1  INIT_DIAGBLK
; P 1892  1    (
; P 1893  1    Diagnostic_Name = EVRMD,
; P 1894  1    Set_Flags_Args = 'QUICK',
; P 1895  1    Set_Event_Args = '',
; P 1896  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1897  1    Test_Start_Text = 'Disk',
; P 1898  1    RunTime = '00:00'
;   1899  1    )
;   1900  1  SUPBLK_END;
;   1901  1
```

```
;   1902  1 %SBTTL 'RCF25 Online Support Block'
;   1903  1
; P 1904  1  INIT_SUPBLK_HEAD
; P 1905  1    (
; P 1906  1    Device_Name = RCF25,
; P 1907  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1908  1    Test_Class = MEN$K_Tstcla_Disk,
; P 1909  1    Testcnfg_No = 1,
; P 1910  1    Diagnostic_Level = '2R'
;   1911  1    ),
; P 1912  1  INIT_TSTCNFG_BLK
; P 1913  1    (
; P 1914  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 1915  1    Diagblk_No = 1
;   1916  1    ),
; P 1917  1  INIT_DIAGBLK
; P 1918  1    (
; P 1919  1    Diagnostic_Name = EVRMD,
; P 1920  1    Set_Flags_Args = 'QUICK',
; P 1921  1    Set_Event_Args = '',
; P 1922  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1923  1    Test_Start_Text = 'Disk',
; P 1924  1    RunTime = '00:00'
;   1925  1    )
;   1926  1  SUPBLK_END;
;   1927  1
```

```
;   1928  1 %SBTTL 'RK06 Online Support Block'
;   1929  1
; P 1930  1  INIT_SUPBLK_HEAD
; P 1931  1    (
; P 1932  1    Device_Name = RK06,
; P 1933  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1934  1    Test_Class = MEN$K_Tstcla_Disk,
; P 1935  1    Testcnfg_No = 1,
; P 1936  1    Diagnostic_Level = '2R'
;   1937  1    ),
; P 1938  1  INIT_TSTCNFG_BLK
; P 1939  1    (
; P 1940  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 1941  1    Diagblk_No = 2
;   1942  1    ),
; P 1943  1  INIT_DIAGBLK
; P 1944  1    (
; P 1945  1    Diagnostic_Name = EVRAA,
; P 1946  1    Set_Flags_Args = 'QUICK',
; P 1947  1    Set_Event_Args = '',
; P 1948  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1949  1    Test_Start_Text = 'Disk',
; P 1950  1    RunTime = '00:00'
;   1951  1    ),
; P 1952  1  INIT_DIAGBLK
; P 1953  1    (
; P 1954  1    Diagnostic_Name = EVRAD,
; P 1955  1    Set_Flags_Args = 'QUICK',
; P 1956  1    Set_Event_Args = '',
; P 1957  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1958  1    Test_Start_Text = 'Disk',
; P 1959  1    RunTime = '00:00'
;   1960  1    )
;   1961  1  SUPBLK_END;
;   1962  1
```

```
;   1963  1 %SBTTL 'RK07 Online Support Block'
;   1964  1
; P 1965  1  INIT_SUPBLK_HEAD
; P 1966  1    (
; P 1967  1    Device_Name = RK07,
; P 1968  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 1969  1    Test_Class = MEN$K_Tstcla_Disk,
; P 1970  1    Testcnfg_No = 1,
; P 1971  1    Diagnostic_Level - '2R'
;   1972  1    ),
; P 1973  1  INIT_TSTCNFG_BLK
; P 1974  1    (
; P 1975  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 1976  1    Diagblk_No = 2
;   1977  1    ),
; P 1978  1  INIT_DIAGBLK
; P 1979  1    (
; P 1980  1    Diagnostic_Name = EVRAA,
; P 1981  1    Set_Flags_Args = 'QUICK',
; P 1982  1    Set_Event_Args = '',
; P 1983  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1984  1    Test_Start_Text = 'Disk',
; P 1985  1    RunTime - '00:00'
;   1986  1    ),
; P 1987  1  INIT_DIAGBLK
; P 1988  1    (
; P 1989  1    Diagnostic_Name - EVRAD,
; P 1990  1    Set_Flags_Args - 'QUICK',
; P 1991  1    Set_Event_Args = '',
; P 1992  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 1993  1    Test_Start_Text = 'Disk',
; P 1994  1    RunTime = '00:00'
;   1995  1    )
;   1996  1  SUPBLK_END;
;   1997  1
```

```
;   1998  1 %SBTTL 'RM03 Online Support Block'
;   1999  1
; P 2000  1  INIT_SUPBLK_HEAD
; P 2001  1    (
; P 2002  1    Device_Name = RM03,
; P 2003  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 2004  1    Test_Class = MEN$K_Tstcla_Disk,
; P 2005  1    Testcnfg_No = 1,
; P 2006  1    Diagnostic_Level = '2R'
;   2007  1    ),
; P 2008  1  INIT_TSTCNFG_BLK
; P 2009  1    (
; P 2010  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 2011  1    Diagblk_No = 2
;   2012  1    ),
; P 2013  1  INIT_DIAGBLK
; P 2014  1    (
; P 2015  1    Diagnostic_Name = EVRAA,
; P 2016  1    Set_Flags_Args = 'QUICK',
; P 2017  1    Set_Event_Args = '',
; P 2018  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2019  1    Test_Start_Text = 'Disk',
; P 2020  1    RunTime = '00:00'
;   2021  1    ),
; P 2022  1  INIT_DIAGBLK
; P 2023  1    (
; P 2024  1    Diagnostic_Name = EVRAD,
; P 2025  1    Set_Flags_Args = 'QUICK',
; P 2026  1    Set_Event_Args = '',
; P 2027  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2028  1    Test_Start_Text = 'Disk',
; P 2029  1    RunTime = '00:00'
;   2030  1    )
;   2031  1  SUPBLK_END;
;   2032  1
```

```
;   2033  1 %SBTTL 'RM05 Online Support Block'
;   2034  1
; P 2035  1  INIT_SUPBLK_HEAD
; P 2036  1    (
; P 2037  1    Device_Name = RM05,
; P 2038  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 2039  1    Test_Class = MEN$K_Tstcla_Disk,
; P 2040  1    Testcnfg_No = 1,
; P 2041  1    Diagnostic_Level = '2R'
;   2042  1    ),
; P 2043  1  INIT_TSTCNFG_BLK
; P 2044  1    (
; P 2045  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 2046  1    Diagblk_No = 2
;   2047  1    ),
; P 2048  1  INIT_DIAGBLK
; P 2049  1    (
; P 2050  1    Diagnostic_Name = EVRAA,
; P 2051  1    Set_Flags_Args = 'QUICK',
; P 2052  1    Set_Event_Args = '',
; P 2053  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2054  1    Test_Start_Text = 'Disk',
; P 2055  1    RunTime = '00:00'
;   2056  1    ),
; P 2057  1  INIT_DIAGBLK
; P 2058  1    (
; P 2059  1    Diagnostic_Name = EVRAD,
; P 2060  1    Set_Flags_Args = 'QUICK',
; P 2061  1    Set_Event_Args = '',
; P 2062  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2063  1    Test_Start_Text = 'Disk',
; P 2064  1    RunTime = '00:00'
;   2065  1    )
;   2066  1  SUPBLK_END;
;   2067  1
```

```
;   2068  1 %SBTTL 'RM80 Online Support Block'
;   2069  1
; P 2070  1  INIT_SUPBLK_HEAD
; P 2071  1    (
; P 2072  1    Device_Name = RM80,
; P 2073  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 2074  1    Test_Class = MEN$K_Tstcla_Disk,
; P 2075  1    Testcnfg_No = 1,
; P 2076  1    Diagnostic_Level = '2R'
;   2077  1    ),
; P 2078  1  INIT_TSTCNFG_BLK
; P 2079  1    (
; P 2080  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 2081  1    Diagblk_No = 2
;   2082  1    ),
; P 2083  1  INIT_DIAGBLK
; P 2084  1    (
; P 2085  1    Diagnostic_Name = EVRAA,
; P 2086  1    Set_Flags_Args = 'QUICK',
; P 2087  1    Set_Event_Args = '',
; P 2088  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2089  1    Test_Start_Text = 'Disk',
; P 2090  1    RunTime = '00:00'
;   2091  1    ),
; P 2092  1  INIT_DIAGBLK
; P 2093  1    (
; P 2094  1    Diagnostic_Name = EVRAD,
; P 2095  1    Set_Flags_Args = 'QUICK',
; P 2096  1    Set_Event_Args = '',
; P 2097  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2098  1    Test_Start_Text = 'Disk',
; P 2099  1    RunTime = '00:00'
;   2100  1    )
;   2101  1  SUPBLK_END;
;   2102  1
```

```
;   2103  1 %SBTTL 'RP05 Online Support Block'
;   2104  1
; P 2105  1  INIT_SUPBLK_HEAD
; P 2106  1    (
; P 2107  1    Device_Name = RK05,
; P 2108  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 2109  1    Test_Class = MEN$K_Tstcla_Disk,
; P 2110  1    Testcnfg_No = 1,
; P 2111  1    Diagnostic_Level = '2R'
;   2112  1    ),
; P 2113  1  INIT_TSTCNFG_BLK
; P 2114  1    (
; P 2115  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 2116  1    Diagblk_No = 2
;   2117  1    ),
; P 2118  1  INIT_DIAGBLK
; P 2119  1    (
; P 2120  1    Diagnostic_Name = EVRAA,
; P 2121  1    Set_Flags_Args = 'QUICK',
; P 2122  1    Set_Event_Args = '',
; P 2123  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2124  1    Test_Start_Text = 'Disk',
; P 2125  1    RunTime = '00:00'
;   2126  1    ),
; P 2127  1  INIT_DIAGBLK
; P 2128  1    (
; P 2129  1    Diagnostic_Name = EVRAD,
; P 2130  1    Set_Flags_Args = 'QUICK',
; P 2131  1    Set_Event_Args = '',
; P 2132  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2133  1    Test_Start_Text = 'Disk',
; P 2134  1    RunTime = '00:00'
;   2135  1    )
;   2136  1  SUPBLK_END;
;   2137  1
```

```
;    2138  1 %SBTTL 'RP06 Online Support Block'
;    2139  1
; P  2140  1  INIT_SUPBLK_HEAD
; P  2141  1    (
; P  2142  1    Device_Name = RP06,
; P  2143  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  2144  1    Test_Class = MEN$K_Tstcla_Disk,
; P  2145  1    Testcnfg_No = 1,
; P  2146  1    Diagnostic_Level = '2R'
;    2147  1    ),
; P  2148  1  INIT_TSTCNFG_BLK
; P  2149  1    (
; P  2150  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  2151  1    Diagblk_No = 2
;    2152  1    ),
; P  2153  1  INIT_DIAGBLK
; P  2154  1    (
; P  2155  1    Diagnostic_Name = EVRAA,
; P  2156  1    Set_Flags_Args = 'QUICK',
; P  2157  1    Set_Event_Args = '',
; P  2158  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  2159  1    Test_Start_Text = 'Disk',
; P  2160  1    RunTime = '00:00'
;    2161  1    ),
; P  2162  1  INIT_DIAGBLK
; P  2163  1    (
; P  2164  1    Diagnostic_Name = EVRAD,
; P  2165  1    Set_Flags_Args = 'QUICK',
; P  2166  1    Set_Event_Args = '',
; P  2167  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  2168  1    Test_Start_Text = 'Disk',
; P  2169  1    RunTime = '00:00'
;    2170  1    )
;    2171  1  SUPBLK_END;
;    2172  1
```

```
;   2173  1 %SBTTL 'RP07 Online Support Block'
;   2174  1
; P 2175  1  INIT_SUPBLK_HEAD
; P 2176  1    (
; P 2177  1    Device_Name = RP07,
; P 2178  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 2179  1    Test_Class = MEN$K_Tstcla_Disk,
; P 2180  1    Testcnfg_No = 1,
; P 2181  1    Diagnostic_Level = '2R'
;   2182  1    ),
; P 2183  1  INIT_TSTCNFG_BLK
; P 2184  1    (
; P 2185  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 2186  1    Diagblk_No = 2
;   2187  1    ),
; P 2188  1  INIT_DIAGBLK
; P 2189  1    (
; P 2190  1    Diagnostic_Name = EVRAA,
; P 2191  1    Set_Flags_Args = 'QUICK',
; P 2192  1    Set_Event_Args = '',
; P 2193  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2194  1    Test_Start_Text = 'Disk',
; P 2195  1    RunTime = '00:00'
;   2196  1    ),
; P 2197  1  INIT_DIAGBLK
; P 2198  1    (
; P 2199  1    Diagnostic_Name = EVRAD,
; P 2200  1    Set_Flags_Args = 'QUICK',
; P 2201  1    Set_Event_Args = '',
; P 2202  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2203  1    Test_Start_Text = 'Disk',
; P 2204  1    RunTime = '00:00'
;   2205  1    )
;   2206  1  SUPBLK_END;
;   2207  1
```

```
;   2208  1 %SBTTL 'RX02 Online Support Block'
;   2209  1
; P 2210  1  INIT_SUPBLK_HEAD
; P 2211  1    (
; P 2212  1    Device_Name = RX02,
; P 2213  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P 2214  1    Test_Class = MEN$K_Tstcla_Disk,
; P 2215  1    Testcnfg_No = 1,
; P 2216  1    Diagnostic_Level = '2R'
;   2217  1    ),
; P 2218  1  INIT_TSTCNFG_BLK
; P 2219  1    (
; P 2220  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 2221  1    Diagblk_No = 2
;   2222  1    ),
; P 2223  1  INIT_DIAGBLK
; P 2224  1    (
; P 2225  1    Diagnostic_Name = EVRAA,
; P 2226  1    Set_Flags_Args = 'QUICK',
; P 2227  1    Set_Event_Args = '',
; P 2228  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2229  1    Test_Start_Text = 'Disk',
; P 2230  1    RunTime = '00:00'
;   2231  1    ),
; P 2232  1  INIT_DIAGBLK
; P 2233  1    (
; P 2234  1    Diagnostic_Name = EVRAD,
; P 2235  1    Set_Flags_Args = 'QUICK',
; P 2236  1    Set_Event_Args = '',
; P 2237  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2238  1    Test_Start_Text = 'Disk',
; P 2239  1    RunTime = '00:00'
;   2240  1    )
;   2241  1  SUPBLK_END;
;   2242  1
```

```
;    2243  1 %SBTTL 'TU58 Online Support Block'
;    2244  1
; P  2245  1  INIT_SUPBLK_HEAD
; P  2246  1    (
; P  2247  1    Device_Name = TU58,
; P  2248  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  2249  1    Test_Class = MEN$K_Tstcla Disk,
; P  2250  1    Testcnfg_No = 1,
; P  2251  1    Diagnostic_Level = '2R'
;    2252  1    ),
; P  2253  1  INIT TSTCNFG_BLK
; P  2254  1    (
; P  2255  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P  2256  1    Diagblk_No = 2
;    2257  1    ),
; P  2258  1  INIT_DIAGBLK
; P  2259  1    (
; P  2260  1    Diagnostic_Name = EVRAA,
; P  2261  1    Set_Flags_Args = 'QUICK',
; P  2262  1    Set_Event_Args = '',
; P  2263  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  2264  1    Test_Start_Text = 'Disk',
; P  2265  1    RunTime = '00:00'
;    2266  1    ),
; P  2267  1  INIT_DIAGBLK
; P  2268  1    (
; P  2269  1    Diagnostic_Name = EVRAD,
; P  2270  1    Set_Flags_Args = 'QUICK',
; P  2271  1    Set_Event_Args = '',
; P  2272  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P  2273  1    Test_Start_Text = 'Disk',
; P  2274  1    RunTime - '00:00'
;    2275  1    )
;    2276  1  SUPBLK_END;
;    2277  1
;    2278  1 %SBTTL 'TU81 Online Support Block'
;    2279  1          ! [03]
; P  2280  1  INIT_SUPBLK_HEAD
; P  2281  1    (
; P  2282  1    Device_Name = TU81,
; P  2283  1    Test_Category = MEN$K_Tstctg_IO_Unit,
; P  2284  1    Test_Class = MEN$K_Tstcla_Tape,
; P  2285  1    Testcnfg_No = 1,
; P  2286  1    Diagnostic_Level = '2R'
;    2287  1    ),
; P  2288  1  INIT_TSTCNFG_BLK
; P  2289  1    (
; P  2290  1    DevTstPrep_Code = MEN$K DevTstPrep 3,
; P  2291  1    Diagblk_No - 1,
; P  2292  1    Status_Scratch Med'a - True
;    2293  1    ),
; P  2294  1  INIT_DIAGBLK
; P  2295  1    (
; P  2296  1    Diagnostic_Name - EVMBA,
; P  2297  1    Set_Flags_Args - 'QUICK',
```

```
;P 2298  1     Set_Event_Args = '',
;P 2299  1     Start_Qualifiers = '/SECTION:DEFAULT',
;P 2300  1     Test_Start_Text = 'Tape',
;P 2301  1     RunTime = '03:45'
;   2302  1     )
;   2303  1  SUPBLK_END;
;   2304  1
;   2305  1
```

```
;   2306  1 %SBTTL 'UNA11 Online Support Block'
;   2307  1
; P 2308  1  INIT_SUPBLK_HEAD
; P 2309  1    (
; P 2310  1    Device_Name = 'UNA11',
; P 2311  1    Test_Category = MEN$K_Tstctg_IO_Controller,
; P 2312  1    Test_Class = MEN$K_Tstcla_Comm,
; P 2313  1    Testcnfg_No = 1,
; P 2314  1    Diagnostic_Level = '2R'
;   2315  1    ),
; P 2316  1  INIT_TSTCNFG_BLK
; P 2317  1    (
; P 2318  1    DevTstPrep_Code = MEN$K_DevTstPrep_Null,
; P 2319  1    Diagblk_No = 1
;   2320  1    ),
; P 2321  1  INIT_DIAGBLK
; P 2322  1    (
; P 2323  1    Diagnostic_Name = EVDWC,
; P 2324  1    Set_Flags_Args = 'QUICK',
; P 2325  1    Set_Event_Args = '',
; P 2326  1    Start_Qualifiers = '/SECTION:DEFAULT',
; P 2327  1    Test_Start_Text = 'NI Exer',
; P 2328  1    RunTime = '00:00'
;   2329  1    )
;   2330  1  SUPBLK_END;
```

```
;   2331  1 END
;   2332  0 ELUDOM


            .TITLE  EVLAA CRD MENU - Functional Test Support
            .IDENT  \V020\

            .PSECT  $$TU58,NOWRT,NOEXE,2

       0000001C  00000     .LONG   28                ;
     20  20  20  20  20  20  20  38  35  55  54  0B  00004 P.ACF:  .ASCII  <11>\TU58       \          ;
        05  00010     .BYTE   5                 ;
        03  00011     .BYTE   3                 ;
        01  00012     .BYTE   1                 ;
             0000  00013     .WORD   0                 ;
           52  32  00015     .ASCII  \2R\                ;
        00  00017     .BYTE   0                 ;
        02  00018     .BYTE   2                 ;
             0000  00019     .WORD   0                 ;
             0006  0001B     .WORD   6                 ;
         41  41  52  56  45  05  0001D     .ASCII  <5>\EVRAA\          ;
         4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
        00  00029     .ASCII  <0>               ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\        ;
            54  4C  00039                 ;
      6B  73  69  44  04  0003B     .ASCII  <4>\Disk\           ;
        30  30  3A  30  30  05  00040     .ASCII  <5>\00:00\          ;
             0006  00046     .WORD   6                 ;
         44  41  52  56  45  05  00048     .ASCII  <5>\EVRAD\          ;
         4B  43  49  55  51  05  0004E     .ASCII  <5>\QUICK\          ;
        00  00054     .ASCII  <0>               ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00055     .ASCII  <16>\/SECTION:DEFAULT\        ;
            54  4C  00064                 ;
      6B  73  69  44  04  00066     .ASCII  <4>\Disk\           ;
        30  30  3A  30  30  05  0006B     .ASCII  <5>\00:00\          ;
           00071    .BLKB   3

            .PSECT  $$RP07,NOWRT,NOEXE,2

       0000001C  00000     .LONG   28                ;
     20  20  20  20  20  20  20  37  30  50  52  0B  00004 P.ACD:  .ASCII  <11>\RP07       \          ;
        05  00010     .BYTE   5                 ;
        03  00011     .BYTE   3                 ;
        01  00012     .BYTE   1                 ;
             0000  00013     .WORD   0                 ;
           52  32  00015     .ASCII  \2R\                ;
        00  00017     .BYTE   0                 ;
        02  00018     .BYTE   2                 ;
             0000  00019     .WORD   0                 ;
             0006  0001B     .WORD   6                 ;
         41  41  52  56  45  05  0001D     .ASCII  <5>\EVRAA\          ;
         4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
        00  00029     .ASCII  <0>               ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\        ;
            54  4C  00039                 ;
```

```
     6B  73  69  44  04  0003B     .ASCII  <4>\Disk\          ;
        30  30  3A  30  30  05  00040     .ASCII  <5>\00:00\          ;
            0006  00046     .WORD   6           ;
        44  41  52  56  45  05  00048     .ASCII  <5>\EVRAD\          ;
        4B  43  49  55  51  05  0004E     .ASCII  <5>\QUICK\          ;
       00  00054     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00055     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00064             ;
     6B  73  69  44  04  00066     .ASCII  <4>\Disk\          ;
        30  30  3A  30  39  05  0006B     .ASCII  <5>\00:00\          ;
           00071    .BLKB   3

           .PSECT  $$RP06,NOWRT,NOEXE,2

        0000001C  00000     .LONG   28          ;
     20  20  20  20  20  20  20  36  30  50  52  0B  00004 P.ACC:  .ASCII  <11>\RP06       \         ;
       05  00010     .BYTE   5           ;
       03  00011     .BYTE   3           ;
       01  00012     .BYTE   1           ;
            0000  00013     .WORD   0           ;
          52  32  00015     .ASCII  \2R\            ;
       00  00017     .BYTE   0           ;
       02  00018     .BYTE   2           ;
            0000  00019     .WORD   0           ;
            0006  0001B     .WORD   6           ;
        41  41  52  56  45  05  0001D     .ASCII  <5>\EVRAA\          ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
       00  00029     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00039             ;
     6B  73  69  44  04  0003B     .ASCII  <4>\Disk\          ;
        30  30  3A  30  30  05  00040     .ASCII  <5>\00:00\          ;
            0006  00046     .WORD   6           ;
        44  41  52  56  45  05  00048     .ASCII  <5>\EVRAD\          ;
        4B  43  49  55  51  05  0004E     .ASCII  <5>\QUICK\          ;
       00  00054     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00055     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00064             ;
     6B  73  69  44  04  00066     .ASCII  <4>\Disk\          ;
        30  30  3A  30  30  05  0006B     .ASCII  <5>\00:00\          ;
           00071    .BLKB   3

           .PSECT  $$RK05,NOWRT,NOEXE,2

        0000001C  00000     .LONG   28          ;
     20  20  20  20  20  20  20  35  30  4B  52  0B  00004 P.ACB:  .ASCII  <11>\RK05       \         ;
       05  00010     .BYTE   5           ;
       03  00011     .BYTE   3           ;
       01  00012     .BYTE   1           ;
            0000  00013     .WORD   0           ;
          52  32  00015     .ASCII  \2R\            ;
       00  00017     .BYTE   0           ;
       02  00018     .BYTE   2           ;
            0000  00019     .WORD   0           ;
            0006  0001B     .WORD   6           ;
```

```
                41  41  52  56  45  05  0001D     .ASCII  <5>\EVRAA\          ;
                4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
              00  00029     .ASCII  <0>                 ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\        ;
                  54  4C  00039                 ;
             6B  73  69  44  04  0003B     .ASCII  <4>\Disk\           ;
                30  30  3A  30  30  05  00040     .ASCII  <5>\00:00\          ;
                    0006  00046     .WORD   6               ;
                44  41  52  56  45  05  00048     .ASCII  <5>\EVRAD\          ;
                4B  43  49  55  51  05  0004E     .ASCII  <5>\QUICK\          ;
              00  00054     .ASCII  <0>                 ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00055     .ASCII  <16>\/SECTION:DEFAULT\        ;
                  54  4C  00064                 ;
             6B  73  69  44  04  00066     .ASCII  <4>\Disk\           ;
                30  30  3A  30  30  05  0006B     .ASCII  <5>\00:00\          ;
                   00071    .BLKB   3

                .PSECT  $$RM80,NOWRT,NOEXE,2

                0000001C  00000     .LONG   28              ;
             20  20  20  20  20  20  20  30  38  4D  52  0B  00004 P.ACA:  .ASCII  <11>\RM80       \         ;
               05  00010     .BYTE   5               ;
               03  00011     .BYTE   3               ;
               01  00012     .BYTE   1               ;
                    0000  00013     .WORD   0               ;
                52  32  00015     .ASCII  \2R\                ;
               00  00017     .BYTE   0               ;
               02  00018     .BYTE   2               ;
                    0000  00019     .WORD   0               ;
                    0006  0001B     .WORD   6               ;
                41  41  52  56  45  05  0001D     .ASCII  <5>\EVRAA\          ;
                4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
              00  00029     .ASCII  <0>                 ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\        ;
                  54  4C  00039                 ;
             6B  73  69  44  04  0003B     .ASCII  <4>\Disk\           ;
                30  30  3A  30  30  05  00040     .ASCII  <5>\00:00\          ;
                    0006  00046     .WORD   6               ;
                44  41  52  56  45  05  00048     .ASCII  <5>\EVRAD\          ;
                4B  43  49  55  51  05  0004E     .ASCII  <5>\QUICK\          ;
              00  00054     .ASCII  <0>                 ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00055     .ASCII  <16>\/SECTION:DEFAULT\        ;
                  54  4C  00064                 ;
             6B  73  69  44  04  00066     .ASCII  <4>\Disk\           ;
                30  30  3A  30  30  05  0006B     .ASCII  <5>\00:00\          ;
                   00071    .BLKB   3

                .PSECT  $$RM05,NOWRT,NOEXE,2

                0000001C  00000     .LONG   28              ;
             20  20  20  20  20  20  20  35  30  4D  52  0B  00004 P.AB7:  .ASCII  <11>\RM05       \         ;
               05  00010     .BYTE   5               ;
               03  00011     .BYTE   3               ;
               01  00012     .BYTE   1               ;
                    0000  00013     .WORD   0               ;
```

```
          52  32  00015     .ASCII  \2R\            ;
       00  00017     .BYTE   0               ;
       02  00018     .BYTE   2               ;
            0000  00019     .WORD   0               ;
            0006  0001B     .WORD   6               ;
        41  41  52  56  45  05  0001D     .ASCII  <5>\EVRAA\          ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
       00  00029     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\      ;
          54  4C  00039               ;
     6B  73  69  44  04  0003B     .ASCII  <4>\Disk\           ;
        30  30  3A  30  30  05  00040     .ASCII  <5>\00:00\          ;
            0006  00046     .WORD   6               ;
        44  41  52  56  45  05  00048     .ASCII  <5>\EVRAD\          ;
        4B  43  49  55  51  05  0004E     .ASCII  <5>\QUICK\          ;
       00  00054     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00055     .ASCII  <16>\/SECTION:DEFAULT\      ;
          54  4C  00064               ;
     6B  73  69  44  04  00066     .ASCII  <4>\Disk\           ;
        30  30  3A  30  30  05  0006B     .ASCII  <5>\00:00\          ;
           00071    .BLKB   3

           .PSECT  $$RM03,NOWRT,NOEXE,2

        0000001C  00000     .LONG   28              ;
     20  20  20  20  20  20  20  33  30  4D  52  0B  00004 P.ABY:  .ASCII  <11>\RM03       \        ;
       05  00010     .BYTE   5               ;
       03  00011     .BYTE   3               ;
       01  00012     .BYTE   1               ;
            0000  00013     .WORD   0               ;
          52  32  00015     .ASCII  \2R\            ;
       00  00017     .BYTE   0               ;
       02  00018     .BYTE   2               ;
            0000  00019     .WORD   0               ;
            0006  0001B     .WORD   6               ;
        41  41  52  56  45  05  0001D     .ASCII  <5>\EVRAA\          ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
       00  00029     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\      ;
          54  4C  00039               ;
     6B  73  69  44  04  0003B     .ASCII  <4>\Disk\           ;
        30  30  3A  30  30  05  00040     .ASCII  <5>\00:00\          ;
            0006  00046     .WORD   6               ;
        44  41  52  56  45  05  00048     .ASCII  <5>\EVRAD\          ;
        4B  43  49  55  51  05  0004E     .ASCII  <5>\QUICK\          ;
       00  00054     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00055     .ASCII  <16>\/SECTION:DEFAULT\      ;
          54  4C  00064               ;
     6B  73  69  44  04  00066     .ASCII  <4>\Disk\           ;
        30  30  3A  30  30  05  0006B     .ASCII  <5>\00:00\          ;
           00071    .BLKB   3

           .PSECT  $$RK06,NOWRT,NOEXE,2

        0000001C  00000     .LONG   28              ;
```

```
     20  20  20  20  20  20  20  36  30  4B  52  0B  00004 P.ABW:  .ASCII  <11>\RK06       \          ;
       05  00010     .BYTE   5           ;
       03  00011     .BYTE   3           ;
       01  00012     .BYTE   1           ;
            0000  00013     .WORD   0           ;
          52  32  00015     .ASCII  \2R\            ;
       00  00017     .BYTE   0           ;
       02  00018     .BYTE   2           ;
            0000  00019     .WORD   0           ;
            0006  0001B     .WORD   6           ;
        41  41  52  56  45  05  0001D     .ASCII  <5>\EVRAA\          ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
       00  00029     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00039               ;
     6B  73  69  44  04  0003B     .ASCII  <4>\Disk\           ;
       30  30  3A  30  30  05  00040     .ASCII  <5>\00:00\          ;
            0006  00046     .WORD   6           ;
        44  41  52  56  45  05  00048     .ASCII  <5>\EVRAD\          ;
        4B  43  49  55  51  05  0004E     .ASCII  <5>\QUICK\          ;
       00  00054     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00055     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00064               ;
     6B  73  69  44  04  00066     .ASCII  <4>\Disk\           ;
       30  30  3A  30  30  05  0006B     .ASCII  <5>\00:00\          ;
          00071     .BLKB   3

          .PSECT  $$DWBUA,NOWRT,NOEXE,2

        0000000F  00000     .LONG   15          ;
     20  20  20  20  20  20  41  55  42  57  44  0B  00004 P.ABM:  .ASCII  <11>\DWBUA      \          ;
       03  00010     .BYTE   3           ;
       0A  00011     .BYTE   10          ;
       01  00012     .BYTE   1           ;
            0000  00013     .WORD   0           ;
          20  33  00015     .ASCII  \3 \            ;
       00  00017     .BYTE   0           ;
       01  00018     .BYTE   1           ;
            0000  00019     .WORD   0           ;
            0006  0001B     .WORD   6           ;
        42  42  43  56  45  05  0001D     .ASCII  <5>\EVCBB\          ;
       00  00023     .ASCII  <0>             ;
       00  00024     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00025     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00034               ;
       00  00036     .ASCII  <0>             ;
        32  32  3A  30  30  05  00037     .ASCII  <5>\00:22\          ;
          0003D     .BLKB   3

          .PSECT  $$BCI750,NOWRT,NOEXE,2

        00000021  00000     .LONG   33          ;
     20  20  20  20  20  30  35  37  49  43  42  0B  00004 P.ABL:  .ASCII  <11>\BCI750     \          ;
       03  00010     .BYTE   3           ;
       0A  00011     .BYTE   10          ;
```

```
       01  00012     .BYTE   1              ;
            0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \           ;
       00  00017     .BYTE   0              ;
       06  00018     .BYTE   6              ;
            0000  00019     .WORD   0              ;
            0006  0001B     .WORD   6              ;
        41  4B  43  56  45  05  0001D     .ASCII  <5>\EVCKA\          ;
       00  00023     .ASCII  <0>            ;
       00  00024     .ASCII  <0>            ;
       00  00025     .ASCII  <0>            ;
       00  00026     .ASCII  <0>            ;
        32  32  3A  30  30  05  00027     .ASCII  <5>\00:22\          ;
            0006  0002D     .WORD   6              ;
        42  4B  43  56  45  05  0002F     .ASCII  <5>\EVCKB\          ;
       00  00035     .ASCII  <0>            ;
       00  00036     .ASCII  <0>            ;
       00  00037     .ASCII  <0>            ;
       00  00038     .ASCII  <0>            ;
        30  30  3A  37  30  05  00039     .ASCII  <5>\07:00\          ;
            0006  0003F     .WORD   6              ;
        43  4B  43  56  45  05  00041     .ASCII  <5>\EVCKC\          ;
       00  00047     .ASCII  <0>            ;
       00  00048     .ASCII  <0>            ;
       00  00049     .ASCII  <0>            ;
       00  0004A     .ASCII  <0>            ;
        30  30  3A  30  31  05  0004B     .ASCII  <5>\10:00\          ;
            0006  00051     .WORD   6              ;
        44  4B  43  56  45  05  00053     .ASCII  <5>\EVCKD\          ;
       00  00059     .ASCII  <0>            ;
       00  0005A     .ASCII  <0>            ;
       00  0005B     .ASCII  <0>            ;
       00  0005C     .ASCII  <0>            ;
        30  30  3A  35  30  05  0005D     .ASCII  <5>\05:00\          ;
            0006  00063     .WORD   6              ;
        45  4B  43  56  45  05  00065     .ASCII  <5>\EVCKE\          ;
       00  0006B     .ASCII  <0>            ;
       00  0006C     .ASCII  <0>            ;
       00  0006D     .ASCII  <0>            ;
       00  0006E     .ASCII  <0>            ;
        30  30  3A  35  30  05  0006F     .ASCII  <5>\05:00\          ;
            0006  00075     .WORD   6              ;
        46  4B  43  56  45  05  00077     .ASCII  <5>\EVCKF\          ;
       00  0007D     .ASCII  <0>            ;
       00  0007E     .ASCII  <0>            ;
       00  0007F     .ASCII  <0>            ;
       00  00080     .ASCII  <0>            ;
        30  30  3A  34  30  05  00081     .ASCII  <5>\04:00\          ;
           00087   .BLKB   1

           .PSECT  $$VS100,NOWRT,NOEXE,2

          0000000B  00000     .LONG   11             ;
    20  20  20  20  20  20  30  30  31  53  56  0B  00004 P.ABK:  .ASCII  <11>\VS100      \          ;
      05  00010     .BYTE   5              ;
```

```
        07  00011     .BYTE   7              ;
        01  00012     .BYTE   1              ;
             0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \             ;
        00  00017     .BYTE   0              ;
        01  00018     .BYTE   1              ;
             0000  00019     .WORD   0              ;
             0006  0001B     .WORD   6              ;
         42  41  57  56  45  05  0001D     .ASCII  <5>\EVWAB\           ;
        00  00023     .ASCII  <0>             ;
        00  00024     .ASCII  <0>             ;
        00  00025     .ASCII  <0>             ;
        00  00026     .ASCII  <0>             ;
         30  30  3A  38  30  05  00027     .ASCII  <5>\08:00\           ;
            0002D    .BLKB   3

            .PSECT  $$TS05,NOWRT,NOEXE,2

          0000000F  00000     .LONG   15             ;
       20  20  20  20  20  20  20  35  30  53  54  0B  00004 P.ABJ:  .ASCII  <11>\TS05       \          ;
        05  00010     .BYTE   5              ;
        04  00011     .BYTE   4              ;
        01  00012     .BYTE   1              ;
             0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \             ;
        03  00017     .BYTE   3              ;
        01  00018     .BYTE   1              ;
             0001  00019     .WORD   1              ;
             0006  0001B     .WORD   6              ;
         49  41  4D  56  45  05  0001D     .ASCII  <5>\EVMAI\           ;
         4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\           ;
        00  00029     .ASCII  <0>             ;
 44  52  43  3A  4E  4F  49  54  43  45  53  2F  0C  0002A     .ASCII  <12>\/SECTION:CRD\         ;
        00  00037     .ASCII  <0>             ;
         35  34  3A  33  30  05  00038     .ASCII  <5>\03:45\           ;
            0003E    .BLKB   2

            .PSECT  $$RX180,NOWRT,NOEXE,2

          00000010  00000     .LONG   16             ;
       20  20  20  20  20  20  30  38  31  58  52  0B  00004 P.ABI:  .ASCII  <11>\RX180      \          ;
        05  00010     .BYTE   5              ;
        03  00011     .BYTE   3              ;
        01  00012     .BYTE   1              ;
             0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \             ;
        07  00017     .BYTE   7              ;
        01  00018     .BYTE   1              ;
             0000  00019     .WORD   0              ;
             0006  0001B     .WORD   6              ;
         42  4E  52  56  45  05  0001D     .ASCII  <5>\EVRNB\           ;
         4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\           ;
        00  00029     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\       ;
          54  4C  00039              ;
```

```
        00  0003B     .ASCII  <0>                ;
        30  30  3A  34  30  05  0003C     .ASCII  <5>\04:00\         ;
            00042   .BLKB   2

            .PSECT  $$RX50,NOWRT,NOEXE,2

          00000010  00000     .LONG   16              ;
      20  20  20  20  20  20  20  30  35  58  52  0B  00004 P.ABH:  .ASCII  <11>\RX50       \         ;
        05  00010     .BYTE   5               ;
        03  00011     .BYTE   3               ;
        01  00012     .BYTE   1               ;
             0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \             ;
        07  00017     .BYTE   7               ;
        01  00018     .BYTE   1               ;
             0000  00019     .WORD   0              ;
             0006  0001B     .WORD   6              ;
        42  4E  52  56  45  05  0001D     .ASCII  <5>\EVRNB\         ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\         ;
        00  00029     .ASCII  <0>                ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\       ;
              54  4C  00039                 ;
        00  0003B     .ASCII  <0>                ;
        30  30  3A  34  30  05  0003C     .ASCII  <5>\04:00\         ;
            00042   .BLKB   2

            .PSECT  $$RX32,NOWRT,NOEXE,2

          00000010  00000     .LONG   16              ;
      20  20  20  20  20  20  20  32  33  58  52  0B  00004 P.ABG:  .ASCII  <11>\RX32       \         ;
        05  00010     .BYTE   5               ;
        03  00011     .BYTE   3               ;
        01  00012     .BYTE   1               ;
             0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \             ;
        07  00017     .BYTE   7               ;
        01  00018     .BYTE   1               ;
             0000  00019     .WORD   0              ;
             0006  0001B     .WORD   6              ;
        42  4E  52  56  45  05  0001D     .ASCII  <5>\EVRNB\         ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\         ;
        00  00029     .ASCII  <0>                ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\       ;
              54  4C  00039                 ;
        00  0003B     .ASCII  <0>                ;
        30  30  3A  34  30  05  0003C     .ASCII  <5>\04:00\         ;
            00042   .BLKB   2

            .PSECT  $$RX02,NOWRT,NOEXE,2

          00000010  00000     .LONG   16              ;
      20  20  20  20  20  20  20  32  30  5P  52  0B  00004 P.ABF:  .ASCII  <11>\RX02       \         ;
        05  00010     .BYTE   5               ;
        03  00011     .BYTE   3               ;
        01  00012     .BYTE   1               ;
```

```
                0000  00013     .WORD   0               ;
              20  33  00015     .ASCII  \3 \              ;
            07  00017     .BYTE   7               ;
            01  00018     .BYTE   1               ;
                0000  00019     .WORD   0               ;
                0006  0001B     .WORD   6               ;
             41  49  52  56  45  05  0001D     .ASCII  <5>\EVRIA\          ;
             4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
            00  00029     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\      ;
              54  4C  00039             ;
            00  0003B     .ASCII  <0>             ;
             30  30  3A  30  30  05  0003C     .ASCII  <5>\00:00\          ;
                00042     .BLKB   2
             0000001C  00044     .LONG   28              ;
         20  20  20  20  20  20  20  32  30  58  52  0B  00048 P.ACE:  .ASCII  <11>\RX02       \        ;
           05  00054     .BYTE   5               ;
           03  00055     .BYTE   3               ;
           01  00056     .BYTE   1               ;
                0000  00057     .WORD   0               ;
              52  32  00059     .ASCII  \2R\              ;
           00  0005B     .BYTE   0               ;
           02  0005C     .BYTE   2               ;
                0000  0005D     .WORD   0               ;
                0006  0005F     .WORD   6               ;
             41  41  52  56  45  05  00061     .ASCII  <5>\EVRAA\          ;
             4B  43  49  55  51  05  00067     .ASCII  <5>\QUICK\          ;
            00  0006D     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0006E     .ASCII  <16>\/SECTION:DEFAULT\      ;
              54  4C  0007D             ;
         6B  73  69  44  04  0007F     .ASCII  <4>\Disk\           ;
             30  30  3A  30  30  05  00084     .ASCII  <5>\00:00\          ;
                0006  0008A     .WORD   6               ;
             44  41  52  56  45  05  0008C     .ASCII  <5>\EVRAD\          ;
             4B  43  49  55  51  05  00092     .ASCII  <5>\QUICK\          ;
            00  00098     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00099     .ASCII  <16>\/SECTION:DEFAULT\      ;
              54  4C  000A8             ;
         6B  73  69  44  04  000AA     .ASCII  <4>\Disk\           ;
             30  30  3A  30  30  05  000AF     .ASCII  <5>\00:00\          ;
                000B5     .BLKB   3

                .PSECT  $$RX31,NOWRT,NOEXE,2

             00000010  00000     .LONG   16              ;
         20  20  20  20  20  20  20  31  33  58  52  0B  00004 P.ABE:  .ASCII  <11>\RX31       \        ;
           05  00010     .BYTE   5               ;
           03  00011     .BYTE   3               ;
           01  00012     .BYTE   1               ;
                0000  00013     .WORD   0               ;
              20  33  00015     .ASCII  \3 \              ;
           07  00017     .BYTE   7               ;
           01  00018     .BYTE   1               ;
                0000  00019     .WORD   0               ;
                0006  0001B     .WORD   6               ;
```

```
                42  4E  52  56  45  05  0001D     .ASCII  <5>\EVRNB\          ;
                4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
              00  00029     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\        ;
                  54  4C  00039                 ;
              00  0003B     .ASCII  <0>           ;
                30  30  3A  34  30  05  0003C     .ASCII  <5>\04:00\          ;
                   00042    .BLKB   2

                   .PSECT  $$RCF25,NOWRT,NOEXE,2

                00000018  00000     .LONG   24              ;
           20  20  20  20  20  20  35  32  46  43  52  0B  00004 P.ABD:  .ASCII  <11>\RCF25      \         ;
              05  00010     .BYTE   5             ;
              03  00011     .BYTE   3             ;
              01  00012     .BYTE   1             ;
                   0000  00013     .WORD   0               ;
                 20  33  00015     .ASCII  \3 \              ;
              06  00017     .BYTE   6             ;
              02  00018     .BYTE   2             ;
                   0000  00019     .WORD   0               ;
                   0006  0001B     .WORD   6               ;
                41  4D  52  56  45  05  0001D     .ASCII  <5>\EVRMA\          ;
                4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
              00  00029     .ASCII  <0>           ;
44  52  43  3A  4E  4F  49  54  43  45  53  2F  0C  0002A     .ASCII  <12>\/SECTION:CRD\        ;
              00  00037     .ASCII  <0>           ;
                30  30  3A  31  30  05  00038     .ASCII  <5>\01:00\          ;
                   0006  0003E     .WORD   6               ;
                42  4D  52  56  45  05  00040     .ASCII  <5>\EVRMB\          ;
                4B  43  49  55  51  05  00046     .ASCII  <5>\QUICK\          ;
              00  0004C     .ASCII  <0>           ;
44  52  43  3A  4E  4F  49  54  43  45  53  2F  0C  0004D     .ASCII  <12>\/SECTION:CRD\        ;
              00  0005A     .ASCII  <0>           ;
                30  30  3A  31  30  05  0005B     .ASCII  <5>\01:00\          ;
                   00061    .BLKB   3
                00000011  00064     .LONG   17              ;
           20  20  20  20  20  20  35  32  46  43  52  0B  00068 P.ABV:  .ASCII  <11>\RCF25      \         ;
              05  00074     .BYTE   5             ;
              03  00075     .BYTE   3             ;
              01  00076     .BYTE   1             ;
                   0000  00077     .WORD   0               ;
                 52  32  00079     .ASCII  \2R\              ;
              00  0007B     .BYTE   0             ;
              01  0007C     .BYTE   1             ;
                   0000  0007D     .WORD   0               ;
                   0006  0007F     .WORD   6               ;
                44  4D  52  56  45  05  00081     .ASCII  <5>\EVRMD\          ;
                4B  43  49  55  51  05  00087     .ASCII  <5>\QUICK\          ;
              00  0008D     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0008E     .ASCII  <16>\/SECTION:DEFAULT\        ;
                  54  4C  0009D                 ;
           6B  73  69  44  04  0009F     .ASCII  <4>\Disk\           ;
              30  30  3A  30  30  05  000A4     .ASCII  <5>\00:00\          ;
                   000AA    .BLKB   2
```

```
                .PSECT  $$RC25,NOWRT,NOEXE,2

             00000019  00000     .LONG   25              ;
          20  20  20  20  20  20  20  35  32  43  52  0B  00004 P.ABC:  .ASCII  <11>\RC25       \          ;
            05  00010     .BYTE   5               ;
            03  00011     .BYTE   3               ;
            01  00012     .BYTE   1               ;
                 0000  00013     .WORD   0              ;
               20  33  00015     .ASCII  \3 \             ;
            06  00017     .BYTE   6               ;
            02  00018     .BYTE   2               ;
                 0000  00019     .WORD   0              ;
                 0006  0001B     .WORD   6              ;
             41  4D  52  56  45  05  0001D     .ASCII  <5>\EVRMA\          ;
             4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
            00  00029     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0002A     .ASCII  <16>\/SECTION:DEFAULT\        ;
               54  4C  00039             ;
            00  0003B     .ASCII  <0>             ;
             30  30  3A  31  30  05  0003C     .ASCII  <5>\01:00\          ;
                 0006  00042     .WORD   6              ;
             42  4D  52  56  45  05  00044     .ASCII  <5>\EVRMB\          ;
             4B  43  49  55  51  05  0004A     .ASCII  <5>\QUICK\          ;
            00  00050     .ASCII  <0>             ;
 44  52  43  3A  4E  4F  49  54  43  45  53  2F  0C  00051     .ASCII  <12>\/SECTION:CRD\         ;
            00  0005E     .ASCII  <0>             ;
             30  30  3A  31  30  05  0005F     .ASCII  <5>\01:00\          ;
               00065     .BLKB   3
             00000011  00068     .LONG   17              ;
          20  20  20  20  20  20  20  35  32  43  52  0B  0006C P.ABU:  .ASCII  <11>\RC25       \          ;
            05  00078     .BYTE   5               ;
            03  00079     .BYTE   3               ;
            01  0007A     .BYTE   1               ;
                 0000  0007B     .WORD   0              ;
               52  32  0007D     .ASCII  \2R\             ;
            00  0007F     .BYTE   0               ;
            01  00080     .BYTE   1               ;
                 0000  00081     .WORD   0              ;
                 0006  00083     .WORD   6              ;
             44  4D  52  56  45  05  00085     .ASCII  <5>\EVRMD\          ;
             4B  43  49  55  51  05  0008B     .ASCII  <5>\QUICK\          ;
            00  00091     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00092     .ASCII  <16>\/SECTION:DEFAULT\        ;
               54  4C  000A1             ;
          6B  73  69  44  04  000A3     .ASCII  <4>\Disk\           ;
             30  30  3A  30  30  05  000A8     .ASCII  <5>\00:00\          ;
               000AE     .BLKB   2

                .PSECT  $$DZ32,NOWRT,NOEXE,2

             0000000C  00000     .LONG   12              ;
          20  20  20  20  20  20  20  32  33  5A  44  0B  00004 P.ABB:  .ASCII  <11>\DZ32       \          ;
            04  00010     .BYTE   4               ;
            07  00011     .BYTE   7               ;
```

```
       01  00012     .BYTE   1              ;
            0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \             ;
       00  00017     .BYTE   0              ;
       01  00018     .BYTE   1              ;
            0000  00019     .WORD   0              ;
            0006  0001B     .WORD   6              ;
        42  41  44  56  45  05  0001D     .ASCII  <5>\EVDAB\           ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\           ;
       00  00029     .ASCII  <0>             ;
       00  0002A     .ASCII  <0>             ;
       00  0002B     .ASCII  <0>             ;
        30  30  3A  36  30  05  0002C     .ASCII  <5>\06:00\           ;
           00032    .BLKB   2

           .PSECT  $$XDZ11,NOWRT,NOEXE,2

         0000000C  00000     .LONG   12             ;
    20  20  20  20  20  20  31  31  5A  44  58  0B  00004 P.ABA:  .ASCII  <11>\XDZ11      \          ;
      04  00010     .BYTE   4              ;
      07  00011     .BYTE   7              ;
      01  00012     .BYTE   1              ;
            0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \             ;
      00  00017     .BYTE   0              ;
      01  00018     .BYTE   1              ;
            0000  00019     .WORD   0              ;
            0006  0001B     .WORD   6              ;
        41  41  44  56  45  05  0001D     .ASCII  <5>\EVDAA\           ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\           ;
      00  00029     .ASCII  <0>             ;
      00  0002A     .ASCII  <0>             ;
      00  0002B     .ASCII  <0>             ;
        30  30  3A  30  30  05  0002C     .ASCII  <5>\00:00\           ;
           00032    .BLKB   2

           .PSECT  $$DUP11,NOWRT,NOEXE,2

         0000000C  00000     .LONG   12             ;
    20  20  20  20  20  20  31  31  50  55  44  0B  00004 P.AAZ:  .ASCII  <11>\DUP11      \          ;
      04  00010     .BYTE   4              ;
      05  00011     .BYTE   5              ;
      01  00012     .BYTE   1              ;
            0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \             ;
      00  00017     .BYTE   0              ;
      01  00018     .BYTE   1              ;
            0000  00019     .WORD   0              ;
            0006  0001B     .WORD   6              ;
        50  55  44  56  45  05  0001D     .ASCII  <5>\EVDUP\           ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\           ;
      00  00029     .ASCII  <0>             ;
      00  0002A     .ASCII  <0>             ;
      00  0002B     .ASCII  <0>             ;
        30  31  3A  30  30  05  0002C     .ASCII  <5>\00:10\           ;
```

```
                00032    .BLKB   2

                .PSECT  $$DMP11,NOWRT,NOEXE,2

              0000000B  00000    .LONG   11                ;
            20  20  20  20  20  20  31  31  50  4D  44  0B  00004 P.AAY:  .ASCII  <11>\DMP11      \          ;
              04  00010    .BYTE   4                ;
              05  00011    .BYTE   5                ;
              01  00012    .BYTE   1                ;
                  0000  00013    .WORD   0                ;
                20  33  00015    .ASCII  \3 \               ;
              00  00017    .BYTE   0                ;
              01  00018    .BYTE   1                ;
                  0000  00019    .WORD   0                ;
                  0006  0001B    .WORD   6                ;
               42  4D  44  56  45  05  0001D    .ASCII  <5>\EVDMB\           ;
              00  00023    .ASCII  <0>              ;
              00  00024    .ASCII  <0>              ;
              00  00025    .ASCII  <0>              ;
              00  00026    .ASCII  <0>              ;
               30  30  3A  31  30  05  00027    .ASCII  <5>\01:00\           ;
                0002D    .BLKB   3

                .PSECT  $$DR11W,NOWRT,NOEXE,2

              00000015  00000    .LONG   21                ;
            20  20  20  20  20  20  57  31  31  52  44  0B  00004 P.AAX:  .ASCII  <11>\DR11W      \          ;
              04  00010    .BYTE   4                ;
              06  00011    .BYTE   6                ;
              01  00012    .BYTE   1                ;
                  0000  00013    .WORD   0                ;
                20  33  00015    .ASCII  \3 \               ;
              00  00017    .BYTE   0                ;
              02  00018    .BYTE   2                ;
                  0000  00019    .WORD   0                ;
                  0006  0001B    .WORD   6                ;
               45  52  44  56  45  05  0001D    .ASCII  <5>\EVDRE\           ;
               4B  43  49  55  51  05  00023    .ASCII  <5>\QUICK\           ;
              00  00029    .ASCII  <0>              ;
              00  0002A    .ASCII  <0>              ;
            31  20  54  52  41  50  06  0002B    .ASCII  <6>\PART 1\           ;
               35  31  3A  30  30  05  00032    .ASCII  <5>\00:15\           ;
                  0006  00038    .WORD   6                ;
               42  52  44  56  45  05  0003A    .ASCII  <5>\EVDRB\           ;
               4B  43  49  55  51  05  00040    .ASCII  <5>\QUICK\           ;
              00  00046    .ASCII  <0>              ;
              00  00047    .ASCII  <0>              ;
            32  20  54  52  41  50  06  00048    .ASCII  <6>\PART 2\           ;
               35  31  3A  30  30  05  0004F    .ASCII  <5>\00:15\           ;
                00055    .BLKB   3
              00000011  00058    .LONG   17                ;
             20  20  20  20  20  20  57  31  31  52  44  0B  0005C P.ABO:  .ASCII  <11>\DR11W      \          ;
              04  00068    .BYTE   4                ;
              06  00069    .BYTE   6                ;
              01  0006A    .BYTE   1                ;
```

```
                0000  0006B     .WORD   0             ;
              20  32   0006D     .ASCII  \2 \            ;
           00  0006F     .BYTE   0             ;
           01  00070     .BYTE   1             ;
                0000  00071     .WORD   0             ;
                0006  00073     .WORD   6             ;
            42  52  44  56  45  05  00075     .ASCII  <5>\EVDRB\          ;
            4B  43  49  55  51  05  0007B     .ASCII  <5>\QUICK\          ;
           00  00081     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00082     .ASCII  <16>\/SECTION:DEFAULT\       ;
              54  4C  00091               ;
            57  31  31  52  44  05  00093     .ASCII  <5>\DR11W\          ;
            30  30  3A  30  30  05  00099     .ASCII  <5>\00:00\          ;
              0009F     .BLKB   1

              .PSECT  $$UNA11,NOWRT,NOEXE,2

            0000000B  00000     .LONG   11            ;
         20  20  20  20  20  20  31  31  41  4E  55  0B  00004 P.AAW:  .ASCII  <11>\UNA11      \         ;
           04  00010     .BYTE   4             ;
           05  00011     .BYTE   5             ;
           01  00012     .BYTE   1             ;
                0000  00013     .WORD   0             ;
              20  33  00015     .ASCII  \3 \            ;
           00  00017     .BYTE   0             ;
           01  00018     .BYTE   1             ;
                0000  00019     .WORD   0             ;
                0006  0001B     .WORD   6             ;
            41  57  44  56  45  05  0001D     .ASCII  <5>\EVDWA\          ;
           00  00023     .ASCII  <0>             ;
           00  00024     .ASCII  <0>             ;
           00  00025     .ASCII  <0>             ;
           00  00026     .ASCII  <0>             ;
            30  30  3A  32  30  05  00027     .ASCII  <5>\02:00\          ;
              0002D     .BLKB   3
            00000012  00030     .LONG   18            ;
         20  20  20  20  20  20  31  31  41  4E  55  0B  00034 P.ACH:  .ASCII  <11>\UNA11      \         ;
           04  00040     .BYTE   4             ;
           05  00041     .BYTE   5             ;
           01  00042     .BYTE   1             ;
                0000  00043     .WORD   0             ;
              52  32  00045     .ASCII  \2R\            ;
           00  00047     .BYTE   0             ;
           01  00048     .BYTE   1             ;
                0000  00049     .WORD   0             ;
                0006  0004B     .WORD   6             ;
            43  57  44  56  45  05  0004D     .ASCII  <5>\EVDWC\          ;
            4B  43  49  55  51  05  00053     .ASCII  <5>\QUICK\          ;
           00  00059     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0005A     .ASCII  <16>\/SECTION:DEFAULT\       ;
              54  4C  00069               ;
           72  65  78  45  20  49  4E  07  0006B     .ASCII  <7>\NI Exer\           ;
            30  30  3A  30  30  05  00073     .ASCII  <5>\00:00\          ;
              00079     .BLKB   3
```

```
            .PSECT  $$DMR11,NOWRT,NOEXE,2

       00000015  00000     .LONG   21              ;
     20  20  20  20  20  20  31  31  52  4D  44  0B  00004 P.AAV:  .ASCII  <11>\DMR11      \          ;
       04  00010     .BYTE   4               ;
       05  00011     .BYTE   5               ;
       01  00012     .BYTE   1               ;
            0000  00013     .WORD   0               ;
          20  33  00015     .ASCII  \3 \              ;
       00  00017     .BYTE   0               ;
       02  00018     .BYTE   2               ;
            0000  00019     .WORD   0               ;
            0006  0001B     .WORD   6               ;
        41  58  44  56  45  05  0001D     .ASCII  <5>\EVDXA\          ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
       00  00029     .ASCII  <0>             ;
       00  0002A     .ASCII  <0>             ;
    31  20  54  52  41  50  06  0002B     .ASCII  <6>\PART 1\          ;
        30  33  3A  30  30  05  00032     .ASCII  <5>\00:30\          ;
            0006  00038     .WORD   6               ;
        41  4D  44  56  45  05  0003A     .ASCII  <5>\EVDMA\          ;
        4B  43  49  55  51  05  00040     .ASCII  <5>\QUICK\          ;
       00  00046     .ASCII  <0>             ;
       00  00047     .ASCII  <0>             ;
    32  20  54  52  41  50  06  00048     .ASCII  <6>\PART 2\          ;
        30  33  3A  30  30  05  0004F     .ASCII  <5>\00:30\          ;
          00055   .BLKB   3

            .PSECT  $$DMF32P,NOWRT,NOEXE,2

       0000000D  00000     .LONG   13              ;
    20  20  20  20  20  50  32  33  46  4D  44  0B  00004 P.AAU:  .ASCII  <11>\DMF32P      \          ;
       04  00010     .BYTE   4               ;
       05  00011     .BYTE   5               ;
       01  00012     .BYTE   1               ;
            0000  00013     .WORD   0               ;
          20  33  00015     .ASCII  \3 \              ;
       00  00017     .BYTE   0               ;
       01  00018     .BYTE   1               ;
            0000  00019     .WORD   0               ;
            0006  0001B     .WORD   6               ;
        44  4C  44  56  45  05  0001D     .ASCII  <5>\EVDLD\          ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
       00  00029     .ASCII  <0>             ;
       00  0002A     .ASCII  <0>             ;
        4F  42  4D  4F  43  05  0002B     .ASCII  <5>\COMBO\          ;
        30  33  3A  30  30  05  00031     .ASCII  <5>\00:30\          ;
          00037   .BLKB   1

            .PSECT  $$DMF32A,NOWRT,NOEXE,2

       0000000D  00000     .LONG   13              ;
    20  20  20  20  20  41  32  33  46  4D  44  0B  00004 P.AAT:  .ASCII  <11>\DMF32A      \          ;
       04  00010     .BYTE   4               ;
       05  00011     .BYTE   5               ;
```

```
       01  00012     .BYTE   1              ;
            0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \               ;
       00  00017     .BYTE   0              ;
       01  00018     .BYTE   1              ;
            0000  00019     .WORD   0              ;
            0006  0001B     .WORD   6              ;
        43  4C  44  56  45  05  0001D     .ASCII  <5>\EVDLC\         ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\         ;
       00  00029     .ASCII  <0>             ;
       00  0002A     .ASCII  <0>             ;
        4F  42  4D  4F  43  05  0002B     .ASCII  <5>\COMBO\         ;
        30  33  3A  30  30  05  00031     .ASCII  <5>\00:30\         ;
           00037   .BLKB   1

           .PSECT  $$DMF32S,NOWRT,NOEXE,2

        0000000D  00000     .LONG   13             ;
    20  20  20  20  20  53  32  33  46  4D  44  0B  00004 P.AAS:  .ASCII  <11>\DMF32S     \          ;
       04  00010     .BYTE   4              ;
       05  00011     .BYTE   5              ;
       01  00012     .BYTE   1              ;
            0000  00013     .WORD   0              ;
           20  33  00015     .ASCII  \3 \               ;
       00  00017     .BYTE   0              ;
       01  00018     .BYTE   1              ;
            0000  00019     .WORD   0              ;
            0006  0001B     .WORD   6              ;
        42  4C  44  56  45  05  0001D     .ASCII  <5>\EVDLB\         ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\         ;
       00  00029     .ASCII  <0>             ;
       00  0002A     .ASCII  <0>             ;
        4F  42  4D  4F  43  05  0002B     .ASCII  <5>\COMBO\         ;
        30  30  3A  31  30  05  00031     .ASCII  <5>\01:00\         ;
           00037   .BLKB   1
        00000011  00038     .LONG   17             ;
    20  20  20  20  20  53  32  33  46  4D  44  0B  0003C P.ABN:  .ASCII  <11>\DMF32S     \          ;
       04  00048     .BYTE   4              ;
       05  00049     .BYTE   5              ;
       01  0004A     .BYTE   1              ;
            0000  0004B     .WORD   0              ;
           52  32  0004D     .ASCII  \2R\               ;
       00  0004F     .BYTE   0              ;
       01  00050     .BYTE   1              ;
            0000  00051     .WORD   0              ;
            0006  00053     .WORD   6              ;
        41  4C  44  56  45  05  00055     .ASCII  <5>\EVDLA\         ;
        4B  43  49  55  51  05  0005B     .ASCII  <5>\QUICK\         ;
       00  00061     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00062     .ASCII  <16>\/SECTION:DEFAULT\       ;
          54  4C  00071              ;
    53  32  33  46  4D  44  06  00073     .ASCII  <6>\DMF32S\          ;
       30  30  3A  30  30  05  0007A     .ASCII  <5>\00:00\         ;

           .PSECT  $$DHU11,NOWRT,NOEXE,2
```

```
                0000000C  00000     .LONG   12                ;
            20  20  20  20  20  20  31  31  55  48  44  0B  00004 P.AAR:  .ASCII  <11>\DHU11       \          ;
              04  00010     .BYTE   4                 ;
              05  00011     .BYTE   5                 ;
              01  00012     .BYTE   1                 ;
                   0000  00013     .WORD   0                 ;
                 20  33  00015     .ASCII  \3 \                ;
              00  00017     .BYTE   0                 ;
              01  00018     .BYTE   1                 ;
                   0000  00019     .WORD   0                 ;
                   0006  0001B     .WORD   6                 ;
               49  41  44  56  45  05  0001D     .ASCII  <5>\EVDAI\          ;
               4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
              00  00029     .ASCII  <0>               ;
              00  0002A     .ASCII  <0>               ;
              00  0002B     .ASCII  <0>               ;
               30  33  3A  30  30  05  0002C     .ASCII  <5>\00:30\          ;
                   00032   .BLKB   2

                .PSECT  $$RA81,NOWRT,NOEXE,2

                0000000F  00000     .LONG   15                ;
            20  20  20  20  20  20  20  31  38  41  52  0B  00004 P.AAQ:  .ASCII  <11>\RA81       \          ;
              05  00010     .BYTE   5                 ;
              03  00011     .BYTE   3                 ;
              01  00012     .BYTE   1                 ;
                   0000  00013     .WORD   0                 ;
                 20  33  00015     .ASCII  \3 \                ;
              04  00017     .BYTE   4                 ;
              01  00018     .BYTE   1                 ;
                   0000  00019     .WORD   0                 ;
                   0006  0001B     .WORD   6                 ;
               41  4C  52  56  45  05  0001D     .ASCII  <5>\EVRLA\          ;
               4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
              00  00029     .ASCII  <0>               ;
        32  31  44  52  43  3A  4E  4F  49  54  43  45  53  2F  0E  0002A     .ASCII  <14>\/SECTION:CRD12\        ;
              00  00039     .ASCII  <0>               ;
               30  33  3A  31  30  05  0003A     .ASCII  <5>\01:30\          ;

                .PSECT  $$RA80,NOWRT,NOEXE,2

                0000000F  00000     .LONG   15                ;
            20  20  20  20  20  20  20  30  38  41  52  0B  00004 P.AAP:  .ASCII  <11>\RA80       \          ;
              05  00010     .BYTE   5                 ;
              03  00011     .BYTE   3                 ;
              01  00012     .BYTE   1                 ;
                   0000  00013     .WORD   0                 ;
                 20  33  00015     .ASCII  \3 \                ;
              04  00017     .BYTE   4                 ;
              01  00018     .BYTE   1                 ;
                   0000  00019     .WORD   0                 ;
                   0006  0001B     .WORD   6                 ;
               41  4C  52  56  45  05  0001D     .ASCII  <5>\EVRLA\          ;
               4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
```

```
      00  00029     .ASCII  <0>                ;
32  31  44  52  43  3A  4E  4F  49  54  43  45  53  2F  0E  0002A     .ASCII  <14>\/SECTION:CRD12\          ;
      00  00039     .ASCII  <0>                ;
       35  34  3A  32  30  05  0003A     .ASCII  <5>\02:45\           ;

           .PSECT  $$RA60,NOWRT,NOEXE,2

      0000000F  00000     .LONG   15           ;
    20  20  20  20  20  20  20  30  36  41  52  0B  00004 P.AAO:  .ASCII  <11>\RA60       \          ;
      05  00010     .BYTE   5                  ;
      03  00011     .BYTE   3                  ;
      01  00012     .BYTE   1                  ;
           0000  00013     .WORD   0                ;
         20  33  00015     .ASCII  \3 \               ;
      01  00017     .BYTE   1                  ;
      01  00018     .BYTE   1                  ;
           0000  00019     .WORD   0                ;
           0006  0001B     .WORD   6                ;
       41  4C  52  56  45  05  0001D     .ASCII  <5>\EVRLA\           ;
       4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\           ;
      00  00029     .ASCII  <0>                ;
32  31  44  52  43  3A  4E  4F  49  54  43  45  53  2F  0E  0002A     .ASCII  <14>\/SECTION:CRD12\          ;
      00  00039     .ASCII  <0>                ;
       30  30  3A  31  30  05  0003A     .ASCII  <5>\01:00\           ;

           .PSECT  $$R80,NOWRT,NOEXE,2

      0000000C  00000     .LONG   12           ;
    20  20  20  20  20  20  20  20  30  38  52  0B  00004 P.AAN:  .ASCII  <11>\R80        \          ;
      05  00010     .BYTE   5                  ;
      03  00011     .BYTE   3                  ;
      01  00012     .BYTE   1                  ;
           0000  00013     .WORD   0                ;
         20  33  00015     .ASCII  \3 \               ;
      04  00017     .BYTE   4                  ;
      01  00018     .BYTE   1                  ;
           0000  00019     .WORD   0                ;
           0006  0001B     .WORD   6                ;
       44  41  52  56  45  05  0001D     .ASCII  <5>\EVRAD\           ;
       4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\           ;
      00  00029     .ASCII  <0>                ;
      00  0002A     .ASCII  <0>                ;
      00  0002B     .ASCII  <0>                ;
       30  30  3A  31  30  05  0002C     .ASCII  <5>\01:00\           ;
          00032   .BLKB   2

           .PSECT  $$RL02,NOWRT,NOEXE,2

      0000000C  00000     .LONG   12           ;
    20  20  20  20  20  20  20  32  30  4C  52  0B  00004 P.AAM:  .ASCII  <11>\RL02       \          ;
      05  00010     .BYTE   5                  ;
      03  00011     .BYTE   3                  ;
      01  00012     .BYTE   1                  ;
           0000  00013     .WORD   0                ;
         20  33  00015     .ASCII  \3 \               ;
```

```
       01  00017     .BYTE   1              ;
       01  00018     .BYTE   1              ;
            0000  00019     .WORD   0              ;
            0006  0001B     .WORD   6              ;
        44  41  52  56  45  05  0001D     .ASCII  <5>\EVRAD\             ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\             ;
       00  00029     .ASCII  <0>             ;
       00  0002A     .ASCII  <0>             ;
       00  0002B     .ASCII  <0>             ;
        35  34  3A  30  30  05  0002C     .ASCII  <5>\00:45\             ;
           00032    .BLKB   2

           .PSECT  $$RK07,NOWRT,NOEXE,2

         0000000C  00000     .LONG   12             ;
     20  20  20  20  20  20  20  37  30  4B  52  0B  00004 P.AAL:   .ASCII  <11>\RK07       \          ;
       05  00010     .BYTE   5              ;
       03  00011     .BYTE   3              ;
       01  00012     .BYTE   1              ;
            0000  00013     .WORD   0              ;
          20  33  00015     .ASCII  \3 \             ;
       01  00017     .BYTE   1              ;
       01  00018     .BYTE   1              ;
            0000  00019     .WORD   0              ;
            0006  0001B     .WORD   6              ;
        44  41  52  56  45  05  0001D     .ASCII  <5>\EVRAD\             ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\             ;
       00  00029     .ASCII  <0>             ;
       00  0002A     .ASCII  <0>             ;
       00  0002B     .ASCII  <0>             ;
        35  34  3A  30  30  05  0002C     .ASCII  <5>\00:45\             ;
           00032    .BLKB   2
         0000001C  00034     .LONG   28             ;
      20  20  20  20  20  20  20  37  30  4B  52  0B  00038 P.ABX:   .ASCII  <11>\RK07       \          ;
        05  00044     .BYTE   5              ;
        03  00045     .BYTE   3              ;
        01  00046     .BYTE   1              ;
             0000  00047     .WORD   0              ;
           52  32  00049     .ASCII  \2R\             ;
        00  0004B     .BYTE   0              ;
        02  0004C     .BYTE   2              ;
             0000  0004D     .WORD   0              ;
             0006  0004F     .WORD   6              ;
         41  41  52  56  45  05  00051     .ASCII  <5>\EVRAA\             ;
         4B  43  49  55  51  05  00057     .ASCII  <5>\QUICK\             ;
        00  0005D     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0005E     .ASCII  <16>\/SECTION:DEFAULT\         ;
          54  4C  0006D             ;
      6B  73  69  44  04  0006F     .ASCII  <4>\Disk\              ;
         30  30  3A  30  30  05  00074     .ASCII  <5>\00:00\             ;
             0006  0007A     .WORD   6              ;
         44  41  52  56  45  05  0007C     .ASCII  <5>\EVRAD\             ;
         4B  43  49  55  51  05  00082     .ASCII  <5>\QUICK\             ;
        00  00088     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00089     .ASCII  <16>\/SECTION:DEFAULT\         ;
```

```
          54  4C  00098                ;
     6B  73  69  44  04  0009A     .ASCII  <4>\Disk\          ;
        30  30  3A  30  30  05  0009F     .ASCII  <5>\00:00\          ;
           000A5    .BLKB   3

           .PSECT  $$RK711,NOWRT,NOEXE,2

        0000002A  00000     .LONG   42              ;
     20  20  20  20  20  20  31  31  37  4B  52  0B  00004 P.AAK:  .ASCII  <11>\RK711      \          ;
       04  00010     .BYTE   4              ;
       03  00011     .BYTE   3              ;
       01  00012     .BYTE   1              ;
            0000  00013     .WORD   0              ;
          20  33  00015     .ASCII  \3 \              ;
       00  00017     .BYTE   0              ;
       05  00018     .BYTE   5              ;
            0000  00019     .WORD   0              ;
            0006  0001B     .WORD   6              ;
        41  45  52  56  45  05  0001D     .ASCII  <5>\EVREA\          ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
       00  00029     .ASCII  <0>             ;
       00  0002A     .ASCII  <0>             ;
     31  20  54  52  41  50  06  0002B     .ASCII  <6>\PART 1\          ;
        30  33  3A  30  30  05  00032     .ASCII  <5>\00:30\          ;
            0006  00038     .WORD   6              ;
        42  45  52  56  45  05  0003A     .ASCII  <5>\EVREB\          ;
        4B  43  49  55  51  05  00040     .ASCII  <5>\QUICK\          ;
       00  00046     .ASCII  <0>             ;
       00  00047     .ASCII  <0>             ;
     32  20  54  52  41  50  06  00048     .ASCII  <6>\PART 2\          ;
        30  33  3A  31  30  05  0004F     .ASCII  <5>\01:30\          ;
            0006  00055     .WORD   6              ;
        43  45  52  56  45  05  00057     .ASCII  <5>\EVREC\          ;
        4B  43  49  55  51  05  0005D     .ASCII  <5>\QUICK\          ;
       00  00063     .ASCII  <0>             ;
       00  00064     .ASCII  <0>             ;
     33  20  54  52  41  50  06  00065     .ASCII  <6>\PART 3\          ;
        30  33  3A  31  30  05  0006C     .ASCII  <5>\01:30\          ;
            0006  00072     .WORD   6              ;
        44  45  52  56  45  05  00074     .ASCII  <5>\EVRED\          ;
        4B  43  49  55  51  05  0007A     .ASCII  <5>\QUICK\          ;
       00  00080     .ASCII  <0>             ;
       00  00081     .ASCII  <0>             ;
     34  20  54  52  41  50  06  00082     .ASCII  <6>\PART 4\          ;
        30  33  3A  30  30  05  00089     .ASCII  <5>\00:30\          ;
            0006  0008F     .WORD   6              ;
        45  45  52  56  45  05  00091     .ASCII  <5>\EVREE\          ;
        4B  43  49  55  51  05  00097     .ASCII  <5>\QUICK\          ;
       00  0009D     .ASCII  <0>             ;
       00  0009E     .ASCII  <0>             ;
     35  20  54  52  41  50  06  0009F     .ASCII  <6>\PART 5\          ;
        35  34  3A  31  30  05  000A6     .ASCII  <5>\01:45\          ;

           .PSECT  $$RK611,NOWRT,NOEXE,2
```

```
        0000002A  00000     .LONG   42                ;
    20  20  20  20  20  20  31  31  36  4B  52  0B  00004 P.AAJ:  .ASCII  <11>\RK611       \          ;
      04  00010     .BYTE   4               ;
      03  00011     .BYTE   3               ;
      01  00012     .BYTE   1               ;
          0000  00013     .WORD   0               ;
        20  33  00015     .ASCII  \3 \              ;
      00  00017     .BYTE   0               ;
      05  00018     .BYTE   5               ;
          0000  00019     .WORD   0               ;
          0006  0001B     .WORD   6               ;
       41  45  52  56  45  05  0001D     .ASCII  <5>\EVREA\          ;
       4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
      00  00029     .ASCII  <0>             ;
      00  0002A     .ASCII  <0>             ;
   31  20  54  52  41  50  06  0002B     .ASCII  <6>\PART 1\           ;
       30  33  3A  30  30  05  00032     .ASCII  <5>\00:30\          ;
          0006  00038     .WORD   6               ;
       42  45  52  56  45  05  0003A     .ASCII  <5>\EVREB\          ;
       4B  43  49  55  51  05  00040     .ASCII  <5>\QUICK\          ;
      00  00046     .ASCII  <0>             ;
      00  00047     .ASCII  <0>             ;
   32  20  54  52  41  50  06  00048     .ASCII  <6>\PART 2\           ;
       30  33  3A  31  30  05  0004F     .ASCII  <5>\01:30\          ;
          0006  00055     .WORD   6               ;
       43  45  52  56  45  05  00057     .ASCII  <5>\EVREC\          ;
       4B  43  49  55  51  05  0005D     .ASCII  <5>\QUICK\          ;
      00  00063     .ASCII  <0>             ;
      00  00064     .ASCII  <0>             ;
   33  20  54  52  41  50  06  00065     .ASCII  <6>\PART 3\           ;
       30  33  3A  31  30  05  0006C     .ASCII  <5>\01:30\          ;
          0006  00072     .WORD   6               ;
       44  45  52  56  45  05  00074     .ASCII  <5>\EVRED\          ;
       4B  43  49  55  51  05  0007A     .ASCII  <5>\QUICK\          ;
      00  00080     .ASCII  <0>             ;
      00  00081     .ASCII  <0>             ;
   34  20  54  52  41  50  06  00082     .ASCII  <6>\PART 4\           ;
       30  33  3A  30  30  05  00089     .ASCII  <5>\00:30\          ;
          0006  0008F     .WORD   6               ;
       45  45  52  56  45  05  00091     .ASCII  <5>\EVREE\          ;
       4B  43  49  55  51  05  00097     .ASCII  <5>\QUICK\          ;
      00  0009D     .ASCII  <0>             ;
      00  0009E     .ASCII  <0>             ;
   35  20  54  52  41  50  06  0009F     .ASCII  <6>\PART 5\           ;
       35  34  3A  31  30  05  000A6     .ASCII  <5>\01:45\          ;

        .PSECT  $$TU81,NOWRT,NOEXE,2

        00000010  00000     .LONG   16                ;
    20  20  20  20  20  20  20  31  38  55  54  0B  00004 P.AAI:  .ASCII  <11>\TU81        \          ;
      05  00010     .BYTE   5               ;
      04  00011     .BYTE   4               ;
      01  00012     .BYTE   1               ;
          0000  00013     .WORD   0               ;
        20  33  00015     .ASCII  \3 \              ;
```

```
                00  00017     .BYTE   0               ;
                01  00018     .BYTE   1               ;
                     0000  00019     .WORD   0                  ;
                     0006  0001B     .WORD   6                  ;
                 42  42  4D  56  45  05  0001D     .ASCII  <5>\EVMBB\          ;
                 4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
                00  00029     .ASCII  <0>                ;
 54  4C  55  41  46  45  44  3A  43  45  53  2F  0C  0002A     .ASCII  <12>\/SEC:DEFAULT\          ;
    31  20  54  52  41  50  06  00037     .ASCII  <6>\PART 1\         ;
                30  33  3A  33  30  05  0003E     .ASCII  <5>\03:30\          ;
                00000011  00044     .LONG   17              ;
             20  20  20  20  20  20  20  31  38  55  54  0B  00048 P.ACG:  .ASCII  <11>\TU81       \          ;
                05  00054     .BYTE   5               ;
                04  00055     .BYTE   4               ;
                01  00056     .BYTE   1               ;
                     0000  00057     .WORD   0                  ;
                   52  32  00059     .ASCII  \2R\                ;
                03  0005B     .BYTE   3               ;
                01  0005C     .BYTE   1               ;
                     0001  0005D     .WORD   1                  ;
                     0006  0005F     .WORD   6                  ;
                 41  42  4D  56  45  05  00061     .ASCII  <5>\EVMBA\          ;
                 4B  43  49  55  51  05  00067     .ASCII  <5>\QUICK\          ;
                00  0006D     .ASCII  <0>                ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0006E     .ASCII  <16>\/SECTION:DEFAULT\          ;
                   54  4C  0007D                   ;
             65  70  61  54  04  0007F     .ASCII  <4>\Tape\           ;
                35  34  3A  33  30  05  00084     .ASCII  <5>\03:45\          ;
                    0008A   .BLKB   2

                .PSECT  $$TU80,NOWRT,NOEXE,2

                0000001A  00000     .LONG   26              ;
             20  20  20  20  20  20  20  30  38  55  54  0B  00004 P.AAH:  .ASCII  <11>\TU80       \          ;
                05  00010     .BYTE   5               ;
                04  00011     .BYTE   4               ;
                01  00012     .BYTE   1               ;
                     0000  00013     .WORD   0                  ;
                   20  33  00015     .ASCII  \3 \                ;
                03  00017     .BYTE   3               ;
                02  00018     .BYTE   2               ;
                     0001  00019     .WORD   1                  ;
                     0006  0001B     .WORD   6                  ;
                 44  42  4D  56  45  05  0001D     .ASCII  <5>\EVMBD\          ;
                 4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
                00  00029     .ASCII  <0>                ;
54  4C  55  41  46  45  44  3A  43  45  53  2F  0C  0002A     .ASCII  <12>\/SEC:DEFAULT\          ;
   31  20  54  52  41  50  06  00037     .ASCII  <6>\PART 1\         ;
                30  33  3A  33  30  05  0003E     .ASCII  <5>\03:30\          ;
                     0006  00044     .WORD   6                  ;
                 45  42  4D  56  45  05  00046     .ASCII  <5>\EVMBE\          ;
                 4B  43  49  55  51  05  0004C     .ASCII  <5>\QUICK\          ;
                00  00052     .ASCII  <0>                ;
  44  52  43  3A  43  45  53  2F  08  00053     .ASCII  <8>\/SEC:CRD\            ;
   32  20  54  52  41  50  06  0005C     .ASCII  <6>\PART 2\         ;
```

```
        30  33  3A  33  30  05  00063     .ASCII  <5>\03:30\          ;
            00069    .BLKB   3

            .PSECT  $$TS11,NOWRT,NOEXE,2

           0000000C  00000     .LONG   12              ;
        20  20  20  20  20  20  20  31  31  53  54  0B  00004 P.AAG:  .ASCII  <11>\TS11       \          ;
          05  00010     .BYTE   5               ;
          04  00011     .BYTE   4               ;
          01  00012     .BYTE   1               ;
               0000  00013     .WORD   0               ;
             20  33  00015     .ASCII  \3 \              ;
          03  00017     .BYTE   3               ;
          01  00018     .BYTE   1               ;
               0001  00019     .WORD   1               ;
               0006  0001B     .WORD   6               ;
           44  41  4D  56  45  05  0001D     .ASCII  <5>\EVMAD\          ;
           4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
          00  00029     .ASCII  <0>             ;
          00  0002A     .ASCII  <0>             ;
          00  0002B     .ASCII  <0>             ;
           30  33  3A  37  30  05  0002C     .ASCII  <5>\07:30\          ;
            00032    .BLKB   2

            .PSECT  $$KA86,NOWRT,NOEXE,2

           00000023  00000     .LONG   35              ;
        20  20  20  20  20  20  20  36  38  41  4B  0B  00004 P.AAF:  .ASCII  <11>\KA86       \          ;
          01  00010     .BYTE   1               ;
          01  00011     .BYTE   1               ;
          01  00012     .BYTE   1               ;
               0000  00013     .WORD   0               ;
             20  33  00015     .ASCII  \3 \              ;
          00  00017     .BYTE   0               ;
          04  00018     .BYTE   4               ;
               0000  00019     .WORD   0               ;
               0006  0001B     .WORD   6               ;
           42  41  4B  56  45  05  0001D     .ASCII  <5>\EVKAB\          ;
           4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
          00  00029     .ASCII  <0>             ;
          00  0002A     .ASCII  <0>             ;
        31  20  54  52  41  50  06  0002B     .ASCII  <6>\PART 1\           ;
           30  30  3A  32  30  05  00032     .ASCII  <5>\02:00\          ;
               0006  00038     .WORD   6               ;
           43  41  4B  56  45  05  0003A     .ASCII  <5>\EVKAC\          ;
           4B  43  49  55  51  05  00040     .ASCII  <5>\QUICK\          ;
          00  00046     .ASCII  <0>             ;
          00  00047     .ASCII  <0>             ;
        32  20  54  52  41  50  06  00048     .ASCII  <6>\PART 2\           ;
           30  30  3A  33  30  05  0004F     .ASCII  <5>\03:00\          ;
               0006  00055     .WORD   6               ;
           44  41  4B  56  45  05  00057     .ASCII  <5>\EVKAD\          ;
           4B  43  49  55  51  05  0005D     .ASCII  <5>\QUICK\          ;
          00  00063     .ASCII  <0>             ;
          00  00064     .ASCII  <0>             ;
```

```
    33  20  54  52  41  50  06  00065     .ASCII  <6>\PART 3\          ;
        35  34  3A  32  30  05  0006C     .ASCII  <5>\02:45\          ;
            0006  00072     .WORD   6           ;
        45  41  4B  56  45  05  00074     .ASCII  <5>\EVKAE\          ;
        4B  43  49  55  51  05  0007A     .ASCII  <5>\QUICK\          ;
       00  00080     .ASCII  <0>          ;
       00  00081     .ASCII  <0>          ;
    34  20  54  52  41  50  06  00082     .ASCII  <6>\PART 4\          ;
        30  30  3A  32  30  05  00089     .ASCII  <5>\02:00\          ;
           0008F    .BLKB   1
        00000029  00090     .LONG   41          ;
     20  20  20  20  20  20  20  36  38  41  4B  0B  00094 P.ABT:  .ASCII  <11>\KA86       \         ;
       01  000A0     .BYTE   1           ;
       01  000A1     .BYTE   1           ;
       01  000A2     .BYTE   1           ;
            0000  000A3     .WORD   0           ;
          52  32  000A5     .ASCII  \2R\            ;
       00  000A7     .BYTE   0           ;
       03  000A8     .BYTE   3           ;
            0000  000A9     .WORD   0           ;
            0006  000AB     .WORD   6           ;
        42  41  4B  56  45  05  000AD     .ASCII  <5>\EVKAB\          ;
        4B  43  49  55  51  05  000B3     .ASCII  <5>\QUICK\          ;
       00  000B9     .ASCII  <0>          ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  000BA     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  000C9               ;
    49  20  74  72  61  50  06  000CB     .ASCII  <6>\Part I\          ;
        35  34  3A  30  30  05  000D2     .ASCII  <5>\00:45\          ;
            0006  000D8     .WORD   6           ;
        43  41  4B  56  45  05  000DA     .ASCII  <5>\EVKAC\          ;
        4B  43  49  55  51  05  000E0     .ASCII  <5>\QUICK\          ;
       00  000E6     .ASCII  <0>          ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  000E7     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  000F6               ;
       49  49  20  74  72  61  50  07  000F8     .ASCII  <7>\Part II\          ;
        30  30  3A  31  30  05  00100     .ASCII  <5>\01:00\          ;
            0006  00106     .WORD   6           ;
        44  41  4B  56  45  05  00108     .ASCII  <5>\EVKAD\          ;
        4B  43  49  55  51  05  0010E     .ASCII  <5>\QUICK\          ;
       00  00114     .ASCII  <0>          ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00115     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00124               ;
   49  49  49  20  74  72  61  50  08  00126     .ASCII  <8>\Part III\          ;
        30  30  3A  32  30  05  0012F     .ASCII  <5>\02:00\          ;
           00135    .BLKB   3

           .PSECT  $$KA820,NOWRT,NOEXE,2

        0000001C  00000     .LONG   28          ;
     20  20  20  20  20  20  30  32  38  41  4B  0B  00004 P.AAE:  .ASCII  <11>\KA820      \         ;
       01  00010     .BYTE   1           ;
       01  00011     .BYTE   1           ;
       01  00012     .BYTE   1           ;
            0000  00013     .WORD   0           ;
          20  33  00015     .ASCII  \3 \            ;
```

```
       00  00017     .BYTE   0            ;
       03  00018     .BYTE   3            ;
            0000  00019     .WORD   0            ;
            0006  0001B     .WORD   6            ;
        42  41  4B  56  45  05  0001D     .ASCII  <5>\EVKAB\           ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\           ;
       00  00029     .ASCII  <0>           ;
       00  0002A     .ASCII  <0>           ;
   31  20  54  52  41  50  06  0002B     .ASCII  <6>\PART 1\           ;
       30  30  3A  32  30  05  00032     .ASCII  <5>\02:00\           ;
            0006  00038     .WORD   6            ;
        43  41  4B  56  45  05  0003A     .ASCII  <5>\EVKAC\           ;
        4B  43  49  55  51  05  00040     .ASCII  <5>\QUICK\           ;
       00  00046     .ASCII  <0>           ;
       00  00047     .ASCII  <0>           ;
   32  20  54  52  41  50  06  00048     .ASCII  <6>\PART 2\           ;
       30  30  3A  33  30  05  0004F     .ASCII  <5>\03:00\           ;
            0006  00055     .WORD   6            ;
        45  41  4B  56  45  05  00057     .ASCII  <5>\EVKAE\           ;
        4B  43  49  55  51  05  0005D     .ASCII  <5>\QUICK\           ;
       00  00063     .ASCII  <0>           ;
       00  00064     .ASCII  <0>           ;
   33  20  54  52  41  50  06  00065     .ASCII  <6>\PART 3\           ;
       30  30  3A  32  30  05  0006C     .ASCII  <5>\02:00\           ;
           00072     .BLKB   2
        0000001D  00074     .LONG   29           ;
    20  20  20  20  20  20  30  32  38  41  4B  0B  00078 P.ABS:  .ASCII  <11>\KA820      \          ;
       01  00084     .BYTE   1            ;
       01  00085     .BYTE   1            ;
       01  00086     .BYTE   1            ;
            0000  00087     .WORD   0            ;
          52  32  00089     .ASCII  \2R\            ;
       00  0008B     .BYTE   0            ;
       02  0008C     .BYTE   2            ;
            0000  0008D     .WORD   0            ;
            0006  0008F     .WORD   6            ;
        42  41  4B  56  45  05  00091     .ASCII  <5>\EVKAB\           ;
        4B  43  49  55  51  05  00097     .ASCII  <5>\QUICK\           ;
       00  0009D     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  0009E     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  000AD                ;
    49  20  74  72  61  50  06  000AF     .ASCII  <6>\Part I\           ;
        35  34  3A  30  30  05  000B6     .ASCII  <5>\00:45\           ;
            0006  000BC     .WORD   6            ;
        43  41  4B  56  45  05  000BE     .ASCII  <5>\EVKAC\           ;
        4B  43  49  55  51  05  000C4     .ASCII  <5>\QUICK\           ;
       00  000CA     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  000CB     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  000DA                ;
       49  49  20  74  72  61  50  07  000DC     .ASCII  <7>\Part II\           ;
        30  30  3A  31  30  05  000E4     .ASCII  <5>\01:00\           ;
           000EA     .BLKB   2

           .PSECT  $$KA780,NOWRT,NOEXE,2
```

```
                00000023  00000     .LONG   35                ;
            20  20  20  20  20  20  30  38  37  41  4B  0B  00004 P.AAD:  .ASCII  <11>\KA780      \          ;
              01  00010     .BYTE   1               ;
              01  00011     .BYTE   1               ;
              01  00012     .BYTE   1               ;
                   0000  00013     .WORD   0                 ;
                 20  33  00015     .ASCII  \3 \                ;
              00  00017     .BYTE   0               ;
              04  00018     .BYTE   4               ;
                   0000  00019     .WORD   0                 ;
                   0006  0001B     .WORD   6                 ;
               42  41  4B  56  45  05  0001D     .ASCII  <5>\EVKAB\          ;
               4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
              00  00029     .ASCII  <0>             ;
              00  0002A     .ASCII  <0>             ;
            31  20  54  52  41  50  06  0002B     .ASCII  <6>\PART 1\          ;
               30  30  3A  32  30  05  00032     .ASCII  <5>\02:00\          ;
                   0006  00038     .WORD   6                 ;
               43  41  4B  56  45  05  0003A     .ASCII  <5>\EVKAC\          ;
               4B  43  49  55  51  05  00040     .ASCII  <5>\QUICK\          ;
              00  00046     .ASCII  <0>             ;
              00  00047     .ASCII  <0>             ;
            32  20  54  52  41  50  06  00048     .ASCII  <6>\PART 2\          ;
               30  30  3A  33  30  05  0004F     .ASCII  <5>\03:00\          ;
                   0006  00055     .WORD   6                 ;
               44  41  4B  56  45  05  00057     .ASCII  <5>\EVKAD\          ;
               4B  43  49  55  51  05  0005D     .ASCII  <5>\QUICK\          ;
              00  00063     .ASCII  <0>             ;
              00  00064     .ASCII  <0>             ;
            33  20  54  52  41  50  06  00065     .ASCII  <6>\PART 3\          ;
               35  34  3A  32  30  05  0006C     .ASCII  <5>\02:45\          ;
                   0006  00072     .WORD   6                 ;
               45  41  4B  56  45  05  00074     .ASCII  <5>\EVKAE\          ;
               4B  43  49  55  51  05  0007A     .ASCII  <5>\QUICK\          ;
              00  00080     .ASCII  <0>             ;
              00  00081     .ASCII  <0>             ;
            34  20  54  52  41  50  06  00082     .ASCII  <6>\PART 4\          ;
               30  30  3A  32  30  05  00089     .ASCII  <5>\02:00\          ;
                0008F     .BLKB   1
                00000029  00090     .LONG   41                ;
             20  20  20  20  20  20  30  38  37  41  4B  0B  00094 P.ABR:  .ASCII  <11>\KA780      \          ;
              01  000A0     .BYTE   1               ;
              01  000A1     .BYTE   1               ;
              01  000A2     .BYTE   1               ;
                   0000  000A3     .WORD   0                 ;
                 52  32  000A5     .ASCII  \2R\                ;
              00  000A7     .BYTE   0               ;
              03  000A8     .BYTE   3               ;
                   0000  000A9     .WORD   0                 ;
                   0006  000AB     .WORD   6                 ;
               42  41  4B  56  45  05  000AD     .ASCII  <5>\EVKAB\          ;
               4B  43  49  55  51  05  000B3     .ASCII  <5>\QUICK\          ;
              00  000B9     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  000BA     .ASCII  <16>\/SECTION:DEFAULT\        ;
                 54  4C  000C9                ;
```

```
    49  20  74  72  61  50  06  000CB     .ASCII  <6>\Part I\          ;
        35  34  3A  30  30  05  000D2     .ASCII  <5>\00:45\          ;
            0006  000D8     .WORD   6           ;
        43  41  4B  56  45  05  000DA     .ASCII  <5>\EVKAC\          ;
        4B  43  49  55  51  05  000E0     .ASCII  <5>\QUICK\          ;
      00  000E6     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  000E7     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  000F6               ;
      49  49  20  74  72  61  50  07  000F8     .ASCII  <7>\Part II\          ;
       30  30  3A  31  30  05  00100     .ASCII  <5>\01:00\          ;
            0006  00106     .WORD   6           ;
       44  41  4B  56  45  05  00108     .ASCII  <5>\EVKAD\          ;
       4B  43  49  55  51  05  0010E     .ASCII  <5>\QUICK\          ;
      00  00114     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00115     .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00124               ;
   49  49  49  20  74  72  61  50  08  00126     .ASCII  <8>\Part III\          ;
       30  30  3A  32  30  05  0012F     .ASCII  <5>\02:00\          ;
           00135   .BLKB   3

           .PSECT  $$KA750,NOWRT,NOEXE,2

        00000023  00000     .LONG   35          ;
     20  20  20  20  20  20  30  35  37  41  4B  0B  00004 P.AAC:  .ASCII  <11>\KA750      \        ;
       01  00010     .BYTE   1           ;
       01  00011     .BYTE   1           ;
       01  00012     .BYTE   1           ;
            0000  00013     .WORD   0           ;
          20  33  00015     .ASCII  \3 \            ;
       00  00017     .BYTE   0           ;
       04  00018     .BYTE   4           ;
            0000  00019     .WORD   0           ;
            0006  0001B     .WORD   6           ;
        42  41  4B  56  45  05  0001D     .ASCII  <5>\EVKAB\          ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
       00  00029     .ASCII  <0>             ;
       00  0002A     .ASCII  <0>             ;
31  20  54  52  41  50  06  0002B     .ASCII  <6>\PART 1\           ;
       30  30  3A  32  30  05  00032     .ASCII  <5>\02:00\          ;
            0006  00038     .WORD   6           ;
       43  41  4B  56  45  05  0003A     .ASCII  <5>\EVKAC\          ;
       4B  43  49  55  51  05  00040     .ASCII  <5>\QUICK\          ;
       00  00046     .ASCII  <0>             ;
       00  00047     .ASCII  <0>             ;
32  20  54  52  41  50  06  00048     .ASCII  <6>\PART 2\           ;
       30  30  3A  33  30  05  0004F     .ASCII  <5>\03:00\          ;
            0006  00055     .WORD   6           ;
       44  41  4B  56  45  05  00057     .ASCII  <5>\EVKAD\          ;
       4B  43  49  55  51  05  0005D     .ASCII  <5>\QUICK\          ;
       00  00063     .ASCII  <0>             ;
       00  00064     .ASCII  <0>             ;
33  20  54  52  41  50  06  00065     .ASCII  <6>\PART 3\           ;
       35  34  3A  32  30  05  0006C     .ASCII  <5>\02:45\          ;
            0006  00072     .WORD   6           ;
       45  41  4B  56  45  05  00074     .ASCII  <5>\EVKAE\          ;
```

```
                4B  43  49  55  51  05  0007A     .ASCII  <5>\QUICK\          ;
              00  00080     .ASCII  <0>           ;
              00  00081     .ASCII  <0>           ;
            34  20  54  52  41  50  06  00082     .ASCII  <6>\PART 4\          ;
                30  30  3A  32  30  05  00089     .ASCII  <5>\02:00\          ;
                    0008F     .BLKB   1
                00000029  00090     .LONG   41          ;
             20  20  20  20  20  20  30  35  37  41  4B  0B  00094 P.ABQ:  .ASCII  <11>\KA750      \          ;
              01  000A0     .BYTE   1           ;
              01  000A1     .BYTE   1           ;
              01  000A2     .BYTE   1           ;
                    0000  000A3     .WORD   0           ;
                  52  32  000A5     .ASCII  \2R\          ;
              00  000A7     .BYTE   0           ;
              03  000A8     .BYTE   3           ;
                    0000  000A9     .WORD   0           ;
                    0006  000AB     .WORD   6           ;
                42  41  4B  56  45  05  000AD     .ASCII  <5>\EVKAB\          ;
                4B  43  49  55  51  05  000B3     .ASCII  <5>\QUICK\          ;
              00  000B9     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  000BA     .ASCII  <16>\/SECTION:DEFAULT\        ;
                  54  4C  000C9             ;
            49  20  74  72  61  50  06  000CB     .ASCII  <6>\Part I\          ;
                35  34  3A  30  30  05  000D2     .ASCII  <5>\00:45\          ;
                    0006  000D8     .WORD   6           ;
                43  41  4B  56  45  05  000DA     .ASCII  <5>\EVKAC\          ;
                4B  43  49  55  51  05  000E0     .ASCII  <5>\QUICK\          ;
              00  000E6     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  000E7     .ASCII  <16>\/SECTION:DEFAULT\        ;
                  54  4C  000F6             ;
              49  49  20  74  72  61  50  07  000F8     .ASCII  <7>\Part II\          ;
                30  30  3A  31  30  05  00100     .ASCII  <5>\01:00\          ;
                    0006  00106     .WORD   6           ;
                44  41  4B  56  45  05  00108     .ASCII  <5>\EVKAD\          ;
                4B  43  49  55  51  05  0010E     .ASCII  <5>\QUICK\          ;
              00  00114     .ASCII  <0>           ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00115     .ASCII  <16>\/SECTION:DEFAULT\        ;
                  54  4C  00124             ;
            49  49  49  20  74  72  61  50  08  00126     .ASCII  <8>\Part III\          ;
                30  30  3A  32  30  05  0012F     .ASCII  <5>\02:00\          ;
                    00135     .BLKB   3

                .PSECT  $$KA730,NOWRT,NOEXE,2

                0000002A  00000     .LONG   42          ;
             20  20  20  20  20  20  30  33  37  41  4B  0B  00004 P.AAB:  .ASCII  <11>\KA730      \          ;
              01  00010     .BYTE   1           ;
              01  00011     .BYTE   1           ;
              01  00012     .BYTE   1           ;
                    0000  00013     .WORD   0           ;
                  20  33  00015     .ASCII  \3 \          ;
              00  00017     .BYTE   0           ;
              05  00018     .BYTE   5           ;
                    0000  00019     .WORD   0           ;
                    0006  0001B     .WORD   6           ;
```

```
        58  41  4B  4E  45  05  0001D     .ASCII  <5>\ENKAX\          ;
        4B  43  49  55  51  05  00023     .ASCII  <5>\QUICK\          ;
     00  00029     .ASCII  <0>             ;
     00  0002A     .ASCII  <0>             ;
  31  20  54  52  41  50  06  0002B     .ASCII  <6>\PART 1\          ;
     35  34  3A  30  30  05  00032     .ASCII  <5>\00:45\          ;
         0006  00038     .WORD   6           ;
     42  41  4B  56  45  05  0003A     .ASCII  <5>\EVKAB\          ;
     4B  43  49  55  51  05  00040     .ASCII  <5>\QUICK\          ;
  00  00046     .ASCII  <0>             ;
  00  00047     .ASCII  <0>             ;
  32  20  54  52  41  50  06  00048     .ASCII  <6>\PART 2\          ;
     30  30  3A  32  30  05  0004F     .ASCII  <5>\02:00\          ;
         0006  00055     .WORD   6           ;
     43  41  4B  56  45  05  00057     .ASCII  <5>\EVKAC\          ;
     4B  43  49  55  51  05  0005D     .ASCII  <5>\QUICK\          ;
  00  00063     .ASCII  <0>             ;
  00  00064     .ASCII  <0>             ;
  33  20  54  52  41  50  06  00065     .ASCII  <6>\PART 3\          ;
     30  30  3A  33  30  05  0006C     .ASCII  <5>\03:00\          ;
         0006  00072     .WORD   6           ;
     44  41  4B  56  45  05  00074     .ASCII  <5>\EVKAD\          ;
     4B  43  49  55  51  05  0007A     .ASCII  <5>\QUICK\          ;
  00  00080     .ASCII  <0>             ;
  00  00081     .ASCII  <0>             ;
  34  20  54  52  41  50  06  00082     .ASCII  <6>\PART 4\          ;
     35  34  3A  32  30  05  00089     .ASCII  <5>\02:45\          ;
         0006  0008F     .WORD   6           ;
     45  41  4B  56  45  05  00091     .ASCII  <5>\EVKAE\          ;
     4B  43  49  55  51  05  00097     .ASCII  <5>\QUICK\          ;
  00  0009D     .ASCII  <0>             ;
  00  0009E     .ASCII  <0>             ;
  35  20  54  52  41  50  06  0009F     .ASCII  <6>\PART 5\          ;
     30  30  3A  32  30  05  000A6     .ASCII  <5>\02:00\          ;
     00000029  000AC     .LONG   41          ;
   20  20  20  20  20  20  30  33  37  41  4B  0B  000B0 P.ABP:  .ASCII  <11>\KA730      \          ;
     01  000BC     .BYTE   1           ;
     01  000BD     .BYTE   1           ;
     01  000BE     .BYTE   1           ;
         0000  000BF     .WORD   0           ;
        52  32  000C1     .ASCII  \2R\            ;
  00  000C3     .BYTE   0           ;
  03  000C4     .BYTE   3           ;
         0000  000C5     .WORD   0           ;
         0006  000C7     .WORD   6           ;
     42  41  4B  56  45  05  000C9     .ASCII  <5>\EVKAB\          ;
     4B  43  49  55  51  05  000CF     .ASCII  <5>\QUICK\          ;
  00  000D5     .ASCII  <0>             ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  000D6     .ASCII  <16>\/SECTION:DEFAULT\       ;
        54  4C  000E5               ;
  49  20  54  52  41  50  06  000E7     .ASCII  <6>\PART I\          ;
     35  34  3A  30  30  05  000EE     .ASCII  <5>\00:45\          ;
         0006  000F4     .WORD   6           ;
     43  41  4B  56  45  05  000F6     .ASCII  <5>\EVKAC\          ;
     4B  43  49  55  51  05  000FC     .ASCII  <5>\QUICK\          ;
```

```
       00  00102    .ASCII  <0>                ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00103    .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00112                ;
       49  49  20  54  52  41  50  07  00114    .ASCII  <7>\PART II\          ;
        30  30  3A  31  30  05  0011C    .ASCII  <5>\01:00\          ;
            0006  00122    .WORD   6           ;
        44  41  4B  56  45  05  00124    .ASCII  <5>\EVKAD\          ;
        4B  43  49  55  51  05  0012A    .ASCII  <5>\QUICK\          ;
       00  00130    .ASCII  <0>                ;
55  41  46  45  44  3A  4E  4F  49  54  43  45  53  2F  10  00131    .ASCII  <16>\/SECTION:DEFAULT\        ;
          54  4C  00140                ;
   49  49  49  20  54  52  41  50  08  00142    .ASCII  <8>\PART III\          ;
        30  30  3A  32  30  05  0014B    .ASCII  <5>\02:00\          ;
           00151    .BLKB   3

           .PSECT  $$$$,NOWRT,NOEXE,2

       00000005  00000    .LONG   5           ;
32  30  56  20  20  3D  20  6E  6F  69  73  72  65  56  13  00004 P.AAA:  .ASCII  <19>\Version =  V02.0   \        ;
     20  20  20  30  2E  00013              ;

          CRD$GA_SUPPORT_DBASE_ID==
                P.AAA
          CRD$GA_SUPBLK_L3__KA730==
                P.AAB
          CRD$GA_SUPBLK_L3__KA750==
                P.AAC
          CRD$GA_SUPBLK_L3__KA780==
                P.AAD
          CRD$GA_SUPBLK_L3__KA820==
                P.AAE
          CRD$GA_SUPBLK_L3__KA86==
                P.AAF
          CRD$GA_SUPBLK_L3__TS11==
                P.AAG
          CRD$GA_SUPBLK_L3__TU80==
                P.AAH
          CRD$GA_SUPBLK_L3__TU81==
                P.AAI
          CRD$GA_SUPBLK_L3__RK611==
                P.AAJ
          CRD$GA_SUPBLK_L3__RK711==
                P.AAK
          CRD$GA_SUPBLK_L3__RK07==
                P.AAL
          CRD$GA_SUPBLK_L3__RL02==
                P.AAM
          CRD$GA_SUPBLK_L3__R80==
                P.AAN
          CRD$GA_SUPBLK_L3__RA60==
                P.AAO
          CRD$GA_SUPBLK_L3__RA80==
                P.AAP
          CRD$GA_SUPBLK L3__RA81==
                P.AAQ
```

```
          CRD$GA_SUPBLK_L3__DHU11==
                P.AAR
          CRD$GA_SUPBLK_L3__DMF32S==
                P.AAS
          CRD$GA_SUPBLK_L3__DMF32A==
                P.AAT
          CRD$GA_SUPBLK_L3__DMF32P==
                P.AAU
          CRD$GA_SUPBLK_L3__DMR11==
                P.AAV
          CRD$GA_SUPBLK_L3__UNA11==
                P.AAW
          CRD$GA_SUPBLK_L3__DR11W==
                P.AAX
          CRD$GA_SUPBLK_L3__DMP11==
                P.AAY
          CRD$GA_SUPBLK_L3__DUP11==
                P.AAZ
          CRD$GA_SUPBLK_L3__XDZ11==
                P.ABA
          CRD$GA_SUPBLK_L3__DZ32==
                P.ABB
          CRD$GA_SUPBLK_L3__RC25==
                P.ABC
          CRD$GA_SUPBLK_L3__RCF25==
                P.ABD
          CRD$GA_SUPBLK_L3__RX31==
                P.ABE
          CRD$GA_SUPBLK_L3__RX02==
                P.ABF
          CRD$GA_SUPBLK_L3__RX32==
                P.ABG
          CRD$GA_SUPBLK_L3__RX50==
                P.ABH
          CRD$GA_SUPBLK_L3__RX180==
                P.ABI
          CRD$GA_SUPBLK_L3__TS05==
                P.ABJ
          CRD$GA_SUPBLK_L3__VS100==
                P.ABK
          CRD$GA_SUPBLK_L3__BCI750==
                P.ABL
          CRD$GA_SUPBLK_L3__DWBUA==
                P.ABM
          CRD$GA_SUPBLK_L2R_DMF32S==
                P.ABN
          CRD$GA_SUPBLK_L2__DR11W==
                P.ABO
          CRD$GA_SUPBLK_L2R_KA730=-
                P.ABP
          CRD$GA_SUPBLK_L2R_KA750==
                P.ABQ
          CRD$GA_SUPBLK_L2R_KA780==
                P.ABR
          CRD$GA_SUPBLK L2R_KA820==
```

```
                P.ABS
          CRD$GA_SUPBLK_L2R_KA86==
                P.ABT
          CRD$GA_SUPBLK_L2R_RC25==
                P.ABU
          CRD$GA_SUPBLK_L2R_RCF25==
                P.ABV
          CRD$GA_SUPBLK_L2R_RK06==
                P.ABW
          CRD$GA_SUPBLK_L2R_RK07==
                P.ABX
          CRD$GA_SUPBLK_L2R_RM03==
                P.ABY
          CRD$GA_SUPBLK_L2R_RM05==
                P.ABZ
          CRD$GA_SUPBLK_L2R_RM80==
                P.ACA
          CRD$GA_SUPBLK_L2R_RK05==
                P.ACB
          CRD$GA_SUPBLK_L2R_RP06==
                P.ACC
          CRD$GA_SUPBLK_L2R_RP07==
                P.ACD
          CRD$GA_SUPBLK_L2R_RX02==
                P.ACE
          CRD$GA_SUPBLK_L2R_TU58=-
                P.ACF
          CRD$GA_SUPBLK_L2R_TU81==
                P.ACG
          CRD$GA_SUPBLK_L2R UNA11==
                P.ACH


;             PSECT SUMMARY
;
; Name     Bytes          Attributes
;
;  $$$$            24  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$KA730          340  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$KA750          312  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$KA780          312  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$KA820          236  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$KA86           312  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$TS11            52  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$TU80           108  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$TU81           140  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RK611          172  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RK711          172  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RK07           168  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RL02            52  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$R80             52  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RA60            64  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RA80            64  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RA81            64  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
;  $$DHU11          52  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$DMF32S        128  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$DMF32A         56  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$DMF32P         56  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$DMR11          88  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$UNA11         124  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$DR11W         160  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$DMP11          48  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$DUP11          52  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$XDZ11          52  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$DZ32           52  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RC25          176  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RCF25         172  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RX31           68  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RX02          184  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RX32           68  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RX50           68  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RX180          68  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$TS05           64  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$VS100          48  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$BCI750        136  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$DWBUA          64  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RK06          116  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RM03          116  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RM05          116  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RM80          116  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RK05          116  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RP06          116  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$RP07          116  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $$TU58          116  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$CRD_LITERALS   Macro    Lib01    88
$EQULST    Macro    Lib02    88       89
$MEN_LITERALS   Macro    Lib01    89
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     88
AUTO$K_EXIT_CONTROL_C  Literal     88
AUTO$K_EXIT_DUMMY_2  Literal     88
AUTO$K_EXIT_DUMMY_3  Literal     88
AUTO$K_EXIT_ERRORS_COMPLETE Literal     88
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     88
AUTO$K_EXIT_INTERNAL_ERROR Literal     88
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     88
AUTO$K_EXIT_LAST_REASON  Literal     88    88
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     88
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     88
AUTO$K_EXIT_NOERRORS_PREP Literal     88
CRD$GA_SUPBLK_L2R_DMF32S GlobBind  1625
CRD$GA_SUPBLK_L2R_KA730  GlobBind  1679
CRD$GA_SUPBLK_L2R_KA750  GlobBind  1722
CRD$GA_SUPBLK_L2R_KA780  GlobBind  1765
CRD$GA_SUPBLK_L2R_KA820  GlobBind  1809
CRD$GA_SUPBLK_L2R_KA86  GlobBind  1843
CRD$GA_SUPBLK_L2R_RC25  GlobBind  1885
CRD$GA_SUPBLK_L2R_RCF25  GlobBind  1911
CRD$GA_SUPBLK_L2R_RK05  GlobBind  2112
CRD$GA_SUPBLK_L2R_RK06  GlobBind  1937
CRD$GA_SUPBLK_L2R_RK07  GlobBind  1972
CRD$GA_SUPBLK_L2R_RM03  GlobBind  2007
CRD$GA_SUPBLK_L2R_RM05  GlobBind  2042
CRD$GA_SUPBLK_L2R_RM80  GlobBind  2077
CRD$GA_SUPBLK_L2R_RP06  GlobBind  2147
CRD$GA_SUPBLK_L2R_RP07  GlobBind  2182
CRD$GA_SUPBLK_L2R_RX02  GlobBind  2217
CRD$GA_SUPBLK_L2R_TU58  GlobBind  2252
CRD$GA_SUPBLK_L2R_TU81  GlobBind  2287
CRD$GA_SUPBLK_L2R_UNA11  GlobBind  2315
CRD$GA_SUPBLK_L2__DR11W  GlobBind  1652
CRD$GA_SUPBLK_L3__BCI750 GlobBind  1490
CRD$GA_SUPBLK_L3__DHU11  GlobBind   906
CRD$GA_SUPBLK_L3__DMF32A GlobBind   959
CRD$GA_SUPBLK_L3__DMF32P GlobBind   983
CRD$GA_SUPBLK_L3__DMF32S GlobBind   933
CRD$GA_SUPBLK_L3__DMP11  GlobBind  1107
CRD$GA_SUPBLK_L3__DMR11  GlobBind  1009
CRD$GA_SUPBLK_L3__DR11W  GlobBind  1071
CRD$GA_SUPBLK_L3__DUP11  GlobBind  1135
CRD$GA_SUPBLK_L3_ DWBUA  GlobBind  1562
CRD$GA_SUPBLK_L3_ DZ32  GlobBind  1196
CRD$GA_SUPBLK_L3__KA730  GlobBind   267
CRD$GA_SUPBLK_L3__KA750  GlobBind   329
CRD$GA_SUPBLK_L3__KA780  GlobBind   382
CRD$GA_SUPBLK_L3__KA820  GlobBind   435
CRD$GA_SUPBLK_L3__KA86  GlobBind   479
CRD$GA_SUPBLK_L3_ R80  GlobBind   805
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$GA_SUPBLK_L3__RA60  GlobBind   831
CRD$GA_SUPBLK_L3__RA80  GlobBind   856
CRD$GA_SUPBLK_L3__RA81  GlobBind   881
CRD$GA_SUPBLK_L3__RC25  GlobBind  1227
CRD$GA_SUPBLK_L3__RCF25  GlobBind  1265
CRD$GA_SUPBLK_L3__RK07  GlobBind   757
CRD$GA_SUPBLK_L3__RK611  GlobBind   628
CRD$GA_SUPBLK_L3__RK711  GlobBind   693
CRD$GA_SUPBLK_L3__RL02  GlobBind   781
CRD$GA_SUPBLK_L3__RX02  GlobBind  1328
CRD$GA_SUPBLK_L3__RX180  GlobBind  1407
CRD$GA_SUPBLK_L3__RX31  GlobBind  1302
CRD$GA_SUPBLK_L3__RX32  GlobBind  1354
CRD$GA_SUPBLK_L3__RX50  GlobBind  1380
CRD$GA_SUPBLK_L3__TS05  GlobBind  1435
CRD$GA_SUPBLK_L3__TS11  GlobBind   532
CRD$GA_SUPBLK_L3__TU80  GlobBind   562
CRD$GA_SUPBLK_L3__TU81  GlobBind   601
CRD$GA_SUPBLK_L3__UNA11  GlobBind  1045
CRD$GA_SUPBLK_L3__VS100  GlobBind  1463
CRD$GA_SUPBLK_L3__XDZ11  GlobBind  1165
CRD$GA_SUPPORT_DBASE_ID  GlobBind   257
CRD$K_ACTION_EQL_DO_NOTHING Literal     88
CRD$K_ACTION_RANGE_ERROR Literal     88
CRD$K_ADVANCE_DIAGNOSTIC Literal     88
CRD$K_ADVANCE_GENERAL_COM Literal     88
CRD$K_ADVANCE_STATE  Literal     88
CRD$K_ALLOCATION_ERROR  Literal     88
CRD$K_ASSIGN_PTABLE_PTRS Literal     88
CRD$K_AUTOSIZER_ERROR  Literal     88
CRD$K_AUTOSIZER_SET_FLAGS Literal     88
CRD$K_BAD_ACTION_NUMBER  Literal     88
CRD$K_BAD_TYPECODE_ERROR Literal     88
CRD$K_BASE_SYSTEM_IDC  Literal     88
CRD$K_BASE_SYSTEM_LESI  Literal     88
CRD$K_BASE_SYSTEM_NULL  Literal     88
CRD$K_BASE_SYSTEM_UDA_0  Literal     88
CRD$K_BASE_SYSTEM_UDA_1  Literal     88
CRD$K_BASE_SYSTEM_UDA_2  Literal     88
CRD$K_BASE_SYSTEM_UDA_3  Literal     88
CRD$K_CRD_INITIALIZATION Literal     88
CRD$K_CREATE_HST  Literal     88
CRD$K_DATA_LENGTH_ERROR  Literal     88
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     88
CRD$K_DDB_BLINK   Literal     88
CRD$K_DDB_FLINK   Literal     88
CRD$K_DDB_LAST_ENTRY  Literal     88
CRD$K_DDB_MEMORY_SIZE  Literal     88
CRD$K_DDB_MENU_DEVICE_NUMB Literal     88
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     88
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     88
CRD$K_DDB_NUMB SUPPORT_TABLES Literal     88
CRD$K_DDB_PTABLE_ENTRY  Literal     88
```

Symbol	Type	Defined	Referenced ...
CRD$K_DDB_STATUS	Literal	88	
CRD$K_DDB_SUPBLK_BLOCK_PTR	Literal	88	
CRD$K_DEVICE_NAME	Literal	88	
CRD$K_DIAGNOSTIC_ERROR	Literal	88	
CRD$K_DIAGNOSTIC_NAME	Literal	88	
CRD$K_DONE_GENERAL_COMS	Literal	88	
CRD$K_DO_NOTHING	Literal	88	
CRD$K_DUMMY_10	Literal	88	
CRD$K_DUMMY_11	Literal	88	
CRD$K_DUMMY_12	Literal	88	
CRD$K_DUMMY_13	Literal	88	
CRD$K_DUMMY_3	Literal	88	
CRD$K_DUMMY_4	Literal	88	
CRD$K_DUMMY_5	Literal	88	
CRD$K_DUMMY_6	Literal	88	
CRD$K_DUMMY_7	Literal	88	
CRD$K_DUMMY_8	Literal	88	
CRD$K_DUMMY_9	Literal	88	
CRD$K_EXIT_CRD	Literal	88	
CRD$K_HOOK_POINT	Literal	88	
CRD$K_HS_INTERNAL_ERROR	Literal	88	
CRD$K_IDC_EVDLB	Literal	88	
CRD$K_IDC_EVDLC	Literal	88	
CRD$K_IDC_EVDLD	Literal	88	
CRD$K_IDC_EVRAD_0	Literal	88	
CRD$K_IDC_EVRAD_1	Literal	88	
CRD$K_IDC_LAST_DIAGNOSTIC	Literal	88	
CRD$K_INTERNAL_ERROR	Literal	88	
CRD$K_LAST_ACTION_ROUTINE	Literal	88	
CRD$K_LAST_ENTRY	Literal	88	
CRD$K_LAST_FUNCTION	Literal	88	
CRD$K_LAST_HOOK_POINT	Literal	88	
CRD$K_LAST_INTERNAL_ERROR	Literal	88	
CRD$K_LAST_TYPE	Literal	88	
CRD$K_LESI_ENWCA	Literal	88	
CRD$K_LESI_EVRMB_0	Literal	88	
CRD$K_LESI_EVRMB_1	Literal	88	
CRD$K_LESI_LAST_DIAGNOSTIC	Literal	88	
CRD$K_LEVEL_4_PASSED	Literal	88	
CRD$K_LOAD_AUTOSIZER	Literal	88	
CRD$K_LOAD_ERROR	Literal	88	
CRD$K_LOAD_GENERAL_COM	Literal	88	
CRD$K_LOAD_LOAD_COM	Literal	88	
CRD$K_LOAD_SELECT_COM	Literal	88	
CRD$K_LOAD_SET_EVENT_COM	Literal	88	
CRD$K_LOAD_SET_FLAGS_COM	Literal	88	
CRD$K_LOAD_START_COM	Literal	88	
CRD$K_LOAD_SUP_FILE	Literal	88	
CRD$K_MM_INTERNAL_ERROR	Literal	88	
CRD$K_NEW_LINE	Literal	88	
CRD$K_PREPARATION_ERROR	Literal	88	
CRD$K_PREPARATION_ERROR_TEXT	Literal	88	
CRD$K_RUNTIME	Literal	88	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_SAME_LINE   Literal     88
CRD$K_SET_EVENT_ARGS  Literal     88
CRD$K_SET_FLAGS_ARGS  Literal     88
CRD$K_SET_UP_DIAGNOSTIC  Literal     88
CRD$K_START   Literal     88
CRD$K_START_AUTOSIZER  Literal     88
CRD$K_START_ERROR  Literal     88
CRD$K_START_QUALIFIERS  Literal     88
CRD$K_STAR_LINE   Literal     88
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal     88
CRD$K_STATE_ADVANCE_GENERAL_COM Literal     88
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal     88
CRD$K_STATE_AUTOSIZER_ERROR Literal     88
CRD$K_STATE_AUTOSIZER_RUNNING Literal     88
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal     88
CRD$K_STATE_DIAGNOSTIC_ERROR Literal     88
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal     88
CRD$K_STATE_DONE_DIAGNOSTICS Literal     88
CRD$K_STATE_DONE_GENERAL_COMS Literal     88
CRD$K_STATE_DUMMY_1  Literal     88
CRD$K_STATE_DUMMY_10  Literal     88
CRD$K_STATE_DUMMY_12  Literal     88
CRD$K_STATE_DUMMY_13  Literal     88
CRD$K_STATE_DUMMY_2  Literal     88
CRD$K_STATE_DUMMY_3  Literal     88
CRD$K_STATE_DUMMY_4  Literal     88
CRD$K_STATE_DUMMY_6  Literal     88
CRD$K_STATE_DUMMY_7  Literal     88
CRD$K_STATE_DUMMY_8  Literal     88
CRD$K_STATE_EXIT_CRD  Literal     88
CRD$K_STATE_INTERNAL_ERROR Literal     88
CRD$K_STATE_LAST_STATE  Literal     88
CRD$K_STATE_LEVEL_4_PASSED Literal     88
CRD$K_STATE_LOAD_AUTOSIZER Literal     88
CRD$K_STATE_LOAD_ERROR  Literal     88
CRD$K_STATE_LOAD_GENERAL_COM Literal     88
CRD$K_STATE_LOAD_LOAD_COM Literal     88
CRD$K_STATE_LOAD_SELECT_COM Literal     88
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     88
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     88
CRD$K_STATE_LOAD_START_COM Literal     88
CRD$K_STATE_PREPARATION_ERROR Literal     88
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     88
CRD$K_STATE_START  Literal     88
CRD$K_STATE_START_AUTOSIZER Literal     88
CRD$K_STATE_START_ERROR  Literal     88
CRD$K_TEST_START_TEXT  Literal     88
CRD$K_TQ_INTERNAL_ERROR  Literal     88
CRD$K_TYPECODE   Literal     88
CRD$K_UDA_0_EVDLB  Literal     88
CRD$K_UDA_0_EVDLC  Literal     88
CRD$K_UDA_0_EVDLD  Literal     88
CRD$K_UDA_0_EVRLA_0  Literal     88
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     88
CRD$K_UDA_1_EVDLB  Literal      88
CRD$K_UDA_1_EVDLC  Literal      88
CRD$K_UDA_1_EVDLD  Literal      88
CRD$K_UDA_1_EVRLA_0  Literal      88
CRD$K_UDA_1_EVRLA_1  Literal      88
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     88
CRD$K_UDA_2_EVDLB  Literal      88
CRD$K_UDA_2_EVDLC  Literal      88
CRD$K_UDA_2_EVDLD  Literal      88
CRD$K_UDA_2_EVRLA_0  Literal      88
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     88
CRD$K_UDA_3_EVDLB  Literal      88
CRD$K_UDA_3_EVDLC  Literal      88
CRD$K_UDA_3_EVDLD  Literal      88
CRD$K_UDA_3_EVRLA_0  Literal      88
CRD$K_UDA_3_EVRLA_1  Literal      88
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     88
DEVICE_NAME   KeyWForm   180   188     193     197
DEVTSTPREP_CODE   KeyWForm   208   212
DIAGBLK_NO   KeyWForm   209   213
DIAGNOSTIC_LEVEL  KeyWForm   186   191     202
DIAGNOSTIC_NAME   KeyWForm   228   235
FALSE    Unbound          184     210
    Literal  Lib01   267     272     329     334     382     387     435     440     479     484
             532     562     601     606     628     633     693     698     757     762
             781     786     805     810     831     836     856     861     881     886
             906     911     933     938     959     964     983     988    1009    1014
            1045    1050    1071    1076    1107    1112    1135    1140    1165    1170
            1196    1201    1227    1232    1265    1270    1302    1308    1328    1334
            1354    1360    1380    1386    1407    1413    1435    1463    1469    1490
            1496    1562    1568    1625    1630    1652    1657    1679    1684    1722
            1727    1765    1770    1809    1814    1843    1848    1885    1890    1911
            1916    1937    1942    1972    1977    2007    2012    2042    2047    2077
            2082    2112    2117    2147    2152    2182    2187    2217    2222    2252
            2257    2287    2315    2320
GET1ST_     Macro    Lib02    88      89
GET2ND_     Unbound          88      89
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal      88
HS$K_ACTION_ADVANCE_GENERAL_COM Literal      88
HS$K_ACTION_ADVANCE_STATE Literal      88
HS$K_ACTION_CHANGE_TABLE Literal      88
HS$K_ACTION_DIAGNOSTIC_ERROR Literal      88
HS$K_ACTION_DONE_GENERAL_COMS Literal      88
HS$K_ACTION_DO_NOTHING  Literal      88
HS$K_ACTION_DUMMY_3  Literal      88
HS$K_ACTION_DUMMY_4  Literal      88
HS$K_ACTION_DUMMY_5  Literal      88
HS$K_ACTION_DUMMY_6  Literal      88
HS$K_ACTION_INIT_HS  Literal      88
HS$K_ACTION_INTERNAL_ERROR Literal      88
HS$K_ACTION_LAST_ACTION  Literal      88
HS$K_ACTION_LOAD_ERROR  Literal      88
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_ACTION_LOAD_GENERAL_COM Literal     88
HS$K_ACTION_LOAD_LOAD_COM Literal     88
HS$K_ACTION_LOAD_SELECT_COM Literal     88
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     88
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     88
HS$K_ACTION_LOAD_START_COM Literal     88
HS$K_ACTION_PREPARATION_ERROR Literal     88
HS$K_ACTION_START_ERROR  Literal     88
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     88
HS$K_STATE_ADVANCE_GENERAL_COM Literal     88
HS$K_STATE_CHANGE_TABLE  Literal     88
HS$K_STATE_DIAGNOSTIC_ERROR Literal     88
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     88
HS$K_STATE_DONE_GENERAL_COMS Literal     88
HS$K_STATE_DUMMY_1  Literal     88
HS$K_STATE_DUMMY_2  Literal     88
HS$K_STATE_DUMMY_3  Literal     88
HS$K_STATE_DUMMY_4  Literal     88
HS$K_STATE_DUMMY_6  Literal     88
HS$K_STATE_INIT_HS  Literal     88
HS$K_STATE_INTERNAL_ERROR Literal     88
HS$K_STATE_LAST_STATE  Literal     88
HS$K_STATE_LOAD_ERROR  Literal     88
HS$K_STATE_LOAD_GENERAL_COM Literal     88
HS$K_STATE_LOAD_LOAD_COM Literal     88
HS$K_STATE_LOAD_SELECT_COM Literal     88
HS$K_STATE_LOAD_SET_EVENT_COM Literal     88
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     88
HS$K_STATE_LOAD_START_COM Literal     88
HS$K_STATE_PREPARATION_ERROR Literal     88
HS$K_STATE_START_ERROR  Literal     88
INIT_DIAGBLK    KeyWMacr    228    273     282     291     300     309     335     344     353     362     388
              397     406     415     441     450     459     485     494     503     512
              539     569     578     607     634     643     652     661     670     699
              708     717     726     735     763     787     811     837     862     887
              912     939     965     989    1015    1024    1051    1077    1086    1113
             1141    1171    1202    1233    1242    1271    1280    1309    1335    1361
             1387    1414    1442    1470    1497    1506    1515    1524    1533    1542
             1569    1631    1658    1685    1694    1703    1728    1737    1746    1771
             1780    1789    1815    1824    1849    1857    1866    1891    1917    1943
             1952    1978    1987    2013    2022    2048    2057    2083    2092    2118
             2127    2153    2162    2188    2197    2223    2232    2258    2267    2294
             2321
INIT_SUPBLK_HEAD  KeyWMacr    180    260     322     375     428     472     525     555     594     621     686
              750     774     798     824     849     874     899     926     952     976
             1002    1038    1064    1100    1128    1158    1189    1220    1258    1295
             1321    1347    1373    1400    1428    1456    1483    1555    1618    1645
             1672    1715    1758    1802    1836    1878    1904    1930    1965    2000
             2035    2070    2105    2140    2175    2210    2245    2280    2308
INIT TSTCNFG_BLK  KeyWMacr    208    268     330     383     436     480     533     563     602     629     694
              758     782     806     832     857     882     907     934     960     984
             1010    1046    1072    1108    1136    1166    1197    1228    1266    1303
             1329    1355    1381    1408    1436    1464    1491    1563    1626    1653
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
             1680    1723    1766    1810    1844    1886    1912    1938    1973    2008
             2043    2078    2113    2148    2183    2218    2253    2288    2316
MEN$GK_TEST_QUEUE  Literal     89
MEN$K_CNTLC_SEL_CONTINUE Literal     89
MEN$K_CNTLC_SEL_EXIT  Literal     89
MEN$K_CNTLC_SEL_FTMENU  Literal     89
MEN$K_CNTLC_SEL_LAST_ENTRY Literal     89
MEN$K_DEVTSTPREP_1  Literal     89   762     786     836
MEN$K_DEVTSTPREP_2  Literal     89
MEN$K_DEVTSTPREP_3  Literal     89   538     568    1441    2293
MEN$K_DEVTSTPREP_4  Literal     89   810     861     886
MEN$K_DEVTSTPREP_5  Literal     89
MEN$K_DEVTSTPREP_6  Literal     89  1232    1270
MEN$K_DEVTSTPREP_7  Literal     89  1308    1334    1360    1386    1413
MEN$K_DEVTSTPREP_NULL  Literal     89   272     334     387     440     484     606     633     698     911     938
              964     988    1014    1050    1076    1112    1140    1170    1201    1469
             1496    1568    1630    1657    1684    1727    1770    1814    1848    1890
             1916    1942    1977    2012    2047    2082    2117    2152    2187    2222
             2257    2320
MEN$K_FTMRET_EXIT  Literal     89
MEN$K_FTMRET_FAILURE  Literal     89
MEN$K_FTMRET_MENU  Literal     89
MEN$K_FTMRET_TEST  Literal     89
MEN$K_FTMSEL_EXIT  Literal     89
MEN$K_FTMSEL_FNCTST_ALL  Literal     89
MEN$K_FTMSEL_LAST_ENTRY  Literal     89
MEN$K_FTMSEL_PREP  Literal     89
MEN$K_FTMSEL_SYSTEM_HDWR Literal     89
MEN$K_FTMSEL_TEST  Literal     89
MEN$K_HS_STATE_TABLE  Literal     88
MEN$K_MM_STATE_TABLE  Literal     88
MEN$K_PREP_ERROR_TEXT_LEN Literal     89
MEN$K_SELDEV_NUMB_START  Literal     89
MEN$K_TMENU_TSTALL_DEVNUMB Literal     89
MEN$K_TQ_STATE_TABLE  Literal     88
MEN$K_TSTCLA_ALL  Literal     89
MEN$K_TSTCLA_CARD  Literal     89
MEN$K_TSTCLA_COMM  Literal     89   906     933     959     983    1009    1045    1107    1135    1625    2315
MEN$K_TSTCLA_CPU  Literal     89   267     329     382     435     479    1679    1722    1765    1809    1843
MEN$K_TSTCLA_DISK  Literal     89   628     693     757     781     805     831     856     881    1227    1265
             1302    1328    1354    1380    1407    1885    1911    1937    1972    2007
             2042    2077    2112    2147    2182    2217    2252
MEN$K_TSTCLA_IO_ADAPTOR  Literal     89  1490    1562
MEN$K_TSTCLA_LAST_ENTRY  Literal     89
MEN$K_TSTCLA_MEMORY  Literal     89
MEN$K_TSTCLA_NULL  Literal     89
MEN$K_TSTCLA_PRINTER  Literal     89
MEN$K_TSTCLA_REAL  Literal     89  1071    1652
MEN$K_TSTCLA_TAPE  Literal     89   532     562     601    1435    2287
MEN$K_TSTCLA_TERM  Literal     89  1165    1196    1463
MEN$K_TSTCTG_CPU  Literal     89   267     329     382     435     479    1679    1722    1765    1809    1843
MEN$K_TSTCTG_IO_ADAPTOR  Literal     89  1490    1562
MEN$K_TSTCTG_IO_CONTROLLER Literal     89   628     693     906     933     959     983    1009    1045    1071    1107
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
                1135    1165    1196    1625    1652    2315
MEN$K_TSTCTG_IO_UNIT  Literal     89    532     562     601     757     781     805     831     856     881    1227
                1265    1302    1328    1354    1380    1407    1435    1463    1885    1911
                1937    1972    2007    2042    2077    2112    2147    2182    2217    2252
                2287
MEN$K_TSTCTG_MEMORY  Literal     89
MEN$K_TSTCTG_NULL  Literal     89
MEN$K_TSTTXT_DEVNAM_SZ  Literal     89
MEN$K_TSTTXT_TSTCLA_SZ  Literal     89
MEN$K_TSTTXT_UUTNAM_SZ  Literal     89
MENU$K_EXIT_AUTOSIZER_ERROR Literal     88
MENU$K_EXIT_INTERNAL_ERROR Literal     88
MENU$K_EXIT_LAST_REASON  Literal     88
MENU$K_EXIT_OPERATOR_ABORT Literal     88
MENU$K_EXIT_OPERATOR_STOP Literal     88
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     88
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     88
MM$K_ACTION_ADVANCE_STATE Literal     88
MM$K_ACTION_AUTOSIZER_ERROR Literal     88
MM$K_ACTION_BUILD_DDBS  Literal     88
MM$K_ACTION_BUILD_HSTS  Literal     88
MM$K_ACTION_CHANGE_TABLE Literal     88
MM$K_ACTION_DONE_INITS  Literal     88
MM$K_ACTION_DO_NOTHING  Literal     88
MM$K_ACTION_DUMMY_3  Literal     88
MM$K_ACTION_DUMMY_4  Literal     88
MM$K_ACTION_DUMMY_5  Literal     88
MM$K_ACTION_DUMMY_6  Literal     88
MM$K_ACTION_DUMMY_7  Literal     88
MM$K_ACTION_DUMMY_8  Literal     88
MM$K_ACTION_INTERNAL_ERROR Literal     88
MM$K_ACTION_LAST_ACTION  Literal     88
MM$K_ACTION_LOAD_AUTOSIZER Literal     88
MM$K_ACTION_MENU_PROMPT  Literal     88
MM$K_ACTION_PRINT_MENU  Literal     88
MM$K_ACTION_PRINT_MENU_HEADING Literal     88
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     88
MM$K_ACTION_START_AUTOSIZER Literal     88
MM$K_STATE_AUTOSIZER_ERROR Literal     88
MM$K_STATE_AUTOSIZER_RUNNING Literal     88
MM$K_STATE_BUILD_DDBS  Literal     88
MM$K_STATE_BUILD_HSTS  Literal     88
MM$K_STATE_CHANGE_TABLE  Literal     88
MM$K_STATE_DONE_INITS  Literal     88
MM$K_STATE_DUMMY_1  Literal     88
MM$K_STATE_DUMMY_2  Literal     88
MM$K_STATE_DUMMY_3  Literal     88
MM$K_STATE_DUMMY_5  Literal     88
MM$K_STATE_DUMMY_7  Literal     88
MM$K_STATE_DUMMY_8  Literal     88
MM$K_STATE_INTERNAL_ERROR Literal     88
MM$K_STATE_LAST_STATE  Literal     88
MM$K_STATE_LOAD_AUTOSIZER Literal     88
```

```
Symbol                            Type  Defined  Referenced ...
-------------------------------- -------- ----- ---------------
MM$K_STATE_MENU_PROMPT  Literal     88
MM$K_STATE_PRINT_MENU  Literal     88
MM$K_STATE_PRINT_MENU_HEADING Literal     88
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     88
MM$K_STATE_START  Literal     88
MM$K_STATE_START_AUTOSIZER Literal     88
NUL2ND_     Macro    Lib02    88      89
RUNTIME     KeyWForm   233   240
SET_EVENT_ARGS    KeyWForm   230   237
SET_FLAGS_ARGS    KeyWForm   229   236
START_QUALIFIERS  KeyWForm   231   238
STATUS_SCRATCH_MEDIA  KeyWForm   210   221
STATUS_SUPBLK_INVALID  KeyWForm   184   201
SUPBLK_END    KeyWMacr   247   318     371     424     468     521     548     587     616     679     744
              772     796     820     846     871     896     921     948     974     998
             1033    1060    1095    1122    1150    1180    1211    1251    1289    1318
             1344    1370    1396    1423    1451    1479    1551    1578    1640    1667
             1712    1755    1798    1833    1875    1900    1926    1961    1996    2031
             2066    2101    2136    2171    2206    2241    2276    2303    2330
SUPPORT_DBASE_ID  KeyWMacr   160   255
TESTCNFG_NO    KeyWForm   183   200
TEST_CATEGORY    KeyWForm   181   198
TEST_CLASS    KeyWForm   182   199
TEST_START_TEXT    KeyWForm   232   239
TQ$K_ACTION_ADVANCE_STATE Literal     88
TQ$K_ACTION_CHANGE_TABLE Literal     88
TQ$K_ACTION_DONE_TEST_QUEUE Literal     88
TQ$K_ACTION_DO_NOTHING  Literal     88
TQ$K_ACTION_DUMMY_3  Literal     88
TQ$K_ACTION_DUMMY_4  Literal     88
TQ$K_ACTION_DUMMY_5  Literal     88
TQ$K_ACTION_GET_DDB  Literal     88
TQ$K_ACTION_INIT_TQ  Literal     88
TQ$K_ACTION_INTERNAL_ERROR Literal     88
TQ$K_ACTION_LAST_ACTION  Literal     88
TQ$K_STATE_CHANGE_TABLE  Literal     88
TQ$K_STATE_DONE_TEST_QUEUE Literal     88
TQ$K_STATE_DUMMY_1  Literal     88
TQ$K_STATE_DUMMY_2  Literal     88
TQ$K_STATE_DUMMY_3  Literal     88
TQ$K_STATE_DUMMY_4  Literal     88
TQ$K_STATE_DUMMY_5  Literal     88
TQ$K_STATE_GET_DDB  Literal     88
TQ$K_STATE_INIT_TQ  Literal     88
TQ$K_STATE_INTERNAL_ERROR Literal     88
TQ$K_STATE_LAST_STATE  Literal     88
TRUE     Literal  Lib01   538     568    1441    2293
VERSION     KeyWForm   160   167     171
```

```
  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ------- -
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V021]EVLAA.B32;1
    3  Module EVLAA
   82  Library #1 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   83  Library #2 SYS$COMMON:[SYSLIB]LIB.L32;2
 2332  Eludom EVLAA


 KEY TO REFERENCE TYPE FLAGS

      . Fetch
      = Store
      c Routine call
      a Address use
      a Indirect use
      e External, external routine, or external literal declaration
      f Forward or forward routine declaration
      h Condition handler enabling
      m Map declaration
      u Undeclare declaration


;     Library Statistics
;
;           -------- Symbols --------      Pages Processing
; File          Total    Loaded   Percent       Mapped Time
;
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486          4          0          62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0        1000          00:02.1




;      COMMAND QUALIFIERS

; BLISS EVLAA.B32/LIST=ENLAA.LIS/CROSS_REFERENCE/DEBUG/OBJECT=ENLAA.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  0 code + 5556 data bytes
; Run Time:    00:41.9
; Elapsed Time:    00:46.0
; Lines/CPU Min:    3340
; Lexemes/CPU-Min: 67541
; Memory Used: 274 pages
; Compilation Complete
```

```
;    0001   0 %TITLE 'BLISS CRD Macros'
;    0002   0 %SBTTL 'For Customer Runnable Diagnostic Development'
;    0003   0 !-------------
;    0004   0 ! Version 1.1 |
;    0005   0 !-------------|
;    0006   0 !++
;    0007   0 !     Copyright (c) 1978, 1982, 1985 BY
;    0008   0 !       DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.
;    0009   0 !
;    0010   0 ! THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND  COPIED
;    0011   0 ! ONLY  IN  ACCORDANCE  WITH  THE  TERMS  OF  SUCH  LICENSE AND WITH THE
;    0012   0 ! INCLUSION OF THE ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE OR  ANY  OTHER
;    0013   0 ! COPIES  THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
;    0014   0 ! OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE  IS  HEREBY
;    0015   0 ! TRANSFERRED.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE  WITHOUT  NOTICE
;    0018   0 ! AND  SHOULD  NOT  BE  CONSTRUED  AS  A COMMITMENT BY DIGITAL EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR  RELIABILITY  OF  ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
;    0023   0 !--
;    0024   0 !++
;    0025   0 ! Facility:
;    0026   0 !
;    0027   0 ! Diagnostic Supervisor
;    0028   0 !
;    0029   0 ! Abstract:
;    0030   0 !
;    0031   0 ! This library contains the literals and macros necessary for development
;    0032   0 ! work on the Customer Runnable Diagnostic package. These macros and
;    0033   0 ! literals are NOT repeated in any other DS library.
;    0034   0 !
;    0035   0 ! Environment:
;    0036   0 !
;    0037   0 ! Not applicable
;    0038   0 !
;    0039   0 ! Author:
;    0040   0 !
;    0041   0 ! Jack Stansbury
;    0042   0 ! John Ciukaj
;    0043   0 !
;    0044   0 ! Creation Date:
;    0045   0 !
;    0046   0 ! April 1, 1982 (April Fools day)
;    0047   0 !
;    0048   0 !
;    0049   0 ! Modified By:
;    0050   0 !
;    0051   0 ! 01 John Ciukaj 4-April-1983 Version 1.2
;    0052   0 !
;    0053   0 ! 02 Scott Cohen 9-June-1983 Version 1.2
;    0054   0 !  For soft console support
;    0055   0 !  - Added routine CRD$Screen_Pause,
```

```
;    0056  0 !  -  macro $SCREEN_Literals for SCREEN$K_ constants,
;    0057  0 !  -  CRD$GT_Clear_Screen,
;    0058  0 !  -  all Auto Summary messages,
;    0059  0 !  - MEN$GT_End_of_List,
;    0060  0 !  - MEN$GT_TM_Pause, etc.
;    0061  0 !  For LCN support,
;    0062  0 !  - Added MEN$GT_DevTstPrep_6
;    0063  0 !
;    0064  0 ! 03 Scott Cohen 23-Sep-1983 Version 1.3
;    0065  0 !  Added new diagnostic ENWCA for PEARL LCN system
;    0066  0 !
;    0067  0 ! 04 Ron Parker 15-OCT-1984
;    0068  0 !  - Added Prep message #7 to $MEN_LITERAL
;    0069  0 !
;    0070  0 ! 05 Amy Iorio 15-JULY-1985
;    0071  0 !  - Deleted crd$gt_clear_screen(_vt52) and put
;    0072  0 !  their equivalent values directly in CRDSTOP.
;    0073  0 !
```

```
;    0074  0 %SBTTL 'EquLst CRD Literals'
;    0075  0
; M  0076  0 MACRO $CRD_Literals =
; M  0077  0  LITERAL
; M  0078  0   $EquLst (CRD$K_, GLOBAL, 0, 1
; M  0079  0      ! These literals define the Function parameter to the CRD$DS_Interface
; M  0080  0      ! routine.
; M  0081  0    ,(Hook_Point) ! A 'hook_point' is a point in the Supervisor from which CRD must be
; M  0082  0      ! called but that has no Print statement (e.g., the initialization of CRD).
; M  0083  0    ,(TypeCode) ! A 'typecode' means that this call to the CRD$DS_Interface routine is the
; M  0084  0      ! result of a Print statement.
; M  0085  0    ,(Last_Function)
; M  0086  0      ! The last function constant. This constant name must ALWAYS be LAST
; M  0087  0      ! in this list
; M  0088  0    );
; M  0089  0
; M  0090  0  LITERAL
; M  0091  0   $EquLst (CRD$K_, GLOBAL, 0, 1
; M  0092  0      ! The literals define the hook points for CRD.
; M  0093  0    ,(CRD_Initialization)
; M  0094  0      ! Time to initialize CRD.
; M  0095  0    ,(Last_Hook_Point)
; M  0096  0      ! The last hook point. This constant name must ALWAYS be LAST
; M  0097  0      ! in this list!
; M  0098  0    );
; M  0099  0
; M  0100  0  !+
; M  0101  0  ! Action routine constants for all four state tables:
; M  0102  0  !-
; M  0103  0
; M  0104  0  LITERAL
; M  0105  0   $EquLst (CRD$K_, GLOBAL, 0, 1
; M  0106  0      ! These literals define the action routine constants for Auto Test.
; M  0107  0    ,(Do_Nothing)
; M  0108  0    ,(Advance_State)
; M  0109  0    ,(Dummy_3)
; M  0110  0    ,(Start)
; M  0111  0    ,(Level_4_Passed)
; M  0112  0    ,(Dummy_4)
; M  0113  0    ,(Load_AutoSizer)
; M  0114  0    ,(AutoSizer Set Flags)
; M  0115  0    ,(Start_AutoSizer)
; M  0116  0    ,(Dummy_5)
; M  0117  0    ,(Assign_PTable Ptrs)
; M  0118  0    ,(Dummy 6)
; M  0119  0    ,(Set_Up_Diagnostic)
; M  0120  0    ,(Dummy_7)
; M  0121  0    ,(Load_General Com)
; M  0122  0    ,(Advance_General Com)
; M  0123  0    ,(Done_Genera  Coms)
; M  0124  0    ,(Dummy_8)
; M  0125  0    ,(Load_Load_Com)
; M  0126  0    ,(Load_Set_Flags_Com)
; M  0127  0    ,(Load_Set_Event_Com)
; M  0128  0    ,(Load_Select_Com)
```

```
; M 0129  0     ,(Load_Start_Com)
; M 0130  0     ,(Dummy_9)
; M 0131  0     ,(Dummy_10)
; M 0132  0     ,(Advance_Diagnustic)
; M 0133  0     ,(Dummy_11)
; M 0134  0     ,(Exit_CRD)
; M 0135  0     ,(Dummy_12)
; M 0136  0     ,(AutoSizer_Error)
; M 0137  0     ,(Internal_Error)
; M 0138  0     ,(Load_Error)
; M 0139  0     ,(Start_Error)
; M 0140  0     ,(Diagnostic_Error)
; M 0141  0     ,(Preparation_Error)
; M 0142  0     ,(Dummy_13)
; M 0143  0     ,(Last_Action_Routine)
; M 0144  0       ! The last action routine constant. This must ALWAYS be the last action
; M 0145  0       ! routine constant in this list. Any additions to this list must be
; M 0146  0       ! made before this name.
; M 0147  0    );
; M 0148  0
; M 0149  0   !+
; M 0150  0   ! VAX-11/730,725 Base System Codes
; M 0151  0   !-
; M 0152  0
; M 0153  0   LITERAL    !          [01]
; M 0154  0    $EquLst (CRD$K_Base_System_, GLOBAL, 0, 1
; M 0155  0       ! These literals define the
; M 0156  0       ! VAX-11/730,725 base system codes
; M 0157  0       !
; M 0158  0     ,(Null)  ! Null system type
; M 0159  0     ,(IDC)  ! System is an IDC Based system.
; M 0160  0       ! Has IDC RB730 Disk controller.
; M 0161  0     ,(LESI)  ! System is a LESI/AZTEC disk based system.
; M 0162  0       ! System does not have IDC.  Has
; M 0163  0       ! a LESI controller at UNIBUS
; M 0164  0       ! I/O fixed CSR address 772150.
; M 0165  0     ,(UDA_0) ! System is a UDA/RA-disk based system.
; M 0166  0       ! System does not have IDC.  Has
; M 0167  0       ! a UDA controller at UNIBUS
; M 0168  0       ! I/O fixed CSR address 772150.  A RA60
; M 0169  0       ! drive IS NOT located at Drive 1.
; M 0170  0       ! A RA80 or 81 is at Drive 0.
; M 0171  0     ,(UDA_1) ! System is a UDA/RA-disk based system.
; M 0172  0       ! System does not have IDC.  Has
; M 0173  0       ! a UDA controller at UNIBUS
; M 0174  0       ! I/O fixed CSR address 772150.  A RA60
; M 0175  0       ! drive IS located at Drive 1.
; M 0176  0       ! A RA80 or 81 is at Drive 0.
; M 0177  0     ,(UDA_2) ! System is a UDA/RA-disk based system.
; M 0178  0       ! System does not have IDC.  Has
; M 0179  0       ! a UDA controller at UNIBUS
; M 0180  0       ! I/O fixed CSR address 772150.  A RA60
; M 0181  0       ! drive IS NOT located at Drive 1.
; M 0182  0       ! A RA60 is at Drive 0.
; M 0183  0     ,(UDA_3) ! System is a UDA/RA-disk based system.
```

```
;  M 0184  0       ! System does not have IDC.  Has
;  M 0185  0       ! a UDA controller at UNIBUS
;  M 0186  0       ! I/O fixed CSR address 772150.  A RA60
;  M 0187  0       ! drive IS located at Drive 1.
;  M 0188  0       ! A RA60 is at Drive 0.
;  M 0189  0     );
;  M 0190  0
;  M 0191  0   LITERAL
;  M 0192  0     $EquLst (MEN$K_, GLOBAL, 0, 1
;  M 0193  0       ! These constants define what state table is currently being used in
;  M 0194  0       ! Menu Test. The variable CRD$GB_Current_State_Table contains one of
;  M 0195  0       ! these three constants.
;  M 0196  0      ,(MM_State_Table)
;  M 0197  0      ,(TQ_State_Table)
;  M 0198  0      ,(HS_State_Table)
;  M 0199  0     );
;  M 0200  0
;  M 0201  0   LITERAL
;  M 0202  0     $EquLst (MM$K_Action_, GLOBAL, 0, 1
;  M 0203  0      ,(Do_Nothing)
;  M 0204  0      ,(Advance_State)
;  M 0205  0       ,(Dummy_3)
;  M 0206  0       ,(Dummy_4)
;  M 0207  0       ,(Dummy_5)
;  M 0208  0      ,(Print_Menu_Heading)
;  M 0209  0      ,(Load_AutoSizer)
;  M 0210  0      ,(Set_Flags_AutoSizer)
;  M 0211  0      ,(Start_AutoSizer)
;  M 0212  0       ,(Dummy_6)
;  M 0213  0      ,(Build_DDBs)
;  M 0214  0      ,(Build_HSTs)
;  M 0215  0       ,(Dummy_7)
;  M 0216  0      ,(Done_Inits)
;  M 0217  0       ,(Dummy_8)
;  M 0218  0      ,(Print_Menu)
;  M 0219  0      ,(Change_Table)
;  M 0220  0      ,(Menu_Prompt)
;  M 0221  0      ,(Internal_Error)
;  M 0222  0      ,(AutoSizer_Error)
;  M 0223  0      ,(Last_Action)
;  M 0224  0       ! The last action routine constant. This must ALWAYS be the last action
;  M 0225  0       ! routine constant in this list. Any additions to this list must be
;  M 0226  0       ! made before this name.
;  M 0227  0     );
;  M 0228  0
;  M 0229  0   LITERAL
;  M 0230  0     $EquLst (TQ$K_Action_, GLOBAL, 0, 1
;  M 0231  0       ! These literals define the action routine constants for the Test Queue.
;  M 0232  0      ,(Do_Nothing)
;  M 0233  0      ,(Advance_State)
;  M 0234  0       ,(Dummy_3)
;  M 0235  0      ,(Init_TQ)
;  M 0236  0      ,(Get_DDB)
;  M 0237  0      ,(Change_Table)
;  M 0238  0       ,(Dummy_4)
```

```
; M 0239  0     ,(Done_Test_Queue)
; M 0240  0      ,(Dummy_5)
; M 0241  0     ,(Internal_Error)
; M 0242  0     ,(Last_Action)
; M 0243  0       ! The last action routine constant. This must ALWAYS be the last action
; M 0244  0       ! routine constant in this list. Any additions to this list must be
; M 0245  0       ! made before this name.
; M 0246  0    );
; M 0247  0
; M 0248  0   LITERAL
; M 0249  0    $EquLst (HS$K_Action_, GLOBAL, 0, 1
; M 0250  0       ! These literals define the action routine constants for the Hardware Support.
; M 0251  0     ,(Do_Nothing)
; M 0252  0     ,(Advance_State)
; M 0253  0      ,(Dummy_3)
; M 0254  0     ,(Init_HS)
; M 0255  0     ,(Advance_Diagnostic)
; M 0256  0     ,(Load_General_Com)
; M 0257  0     ,(Advance_General_Com)
; M 0258  0     ,(Done_General_Coms)
; M 0259  0      ,(Dummy_4)
; M 0260  0     ,(Load_Load_Com)
; M 0261  0     ,(Load_Set_Flags_Com)
; M 0262  0     ,(Load_Set_Event_Com)
; M 0263  0     ,(Load_Select_Com)
; M 0264  0     ,(Load_Start_Com)
; M 0265  0      ,(Dummy_5)
; M 0266  0      ,(Dummy_6)
; M 0267  0     ,(Change_Table)
; M 0268  0     ,(Load_Error)
; M 0269  0     ,(Start_Error)
; M 0270  0     ,(Diagnostic_Error)
; M 0271  0     ,(Preparation_Error)
; M 0272  0     ,(Internal_Error)
; M 0273  0     ,(Last_Action)
; M 0274  0       ! The last action routine constant. This must ALWAYS be the last action
; M 0275  0       ! routine constant in this list. Any additions to this list must be
; M 0276  0       ! made before this name.
; M 0277  0    );
; M 0278  0
; M 0279  0   !+
; M 0280  0   ! These literals define the state numbers for all four state tables:
; M 0281  0   !-
; M 0282  0
; M 0283  0   LITERAL
; M 0284  0    $EquLst (CRD$K_State_, GLOBAL, 0, 1
; M 0285  0       ! These constants define the values used to access the columns in the
; M 0286  0       ! CRD$GW_State_Table. These are the various states that CRD-AutoTest can
; M 0287  0       ! be in at any given time.
; M 0288  0     ,(Dummy_1)
; M 0289  0     ,(Dummy_2)
; M 0290  0     ,(Dummy_3)
; M 0291  0     ,(Start)
; M 0292  0     ,(Level_4_Passed)
; M 0293  0     ,(Dummy_4)
```

```
;  M 0294  0     ,(Load_AutoSizer)
;  M 0295  0     ,(AutoSizer_Set_Flags)
;  M 0296  0     ,(Start_AutoSizer)
;  M 0297  0     ,(AutoSizer_Running)
;  M 0298  0     ,(Assign_PTable_Ptrs)
;  M 0299  0     ,(Dummy_6)
;  M 0300  0     ,(Set_Up_Diagnostic)
;  M 0301  0     ,(Dummy_7)
;  M 0302  0     ,(Load_General_Com)
;  M 0303  0     ,(Advance_General_Com)
;  M 0304  0     ,(Done_General_Coms)
;  M 0305  0     ,(Dummy_8)
;  M 0306  0     ,(Load_Load_Com)
;  M 0307  0     ,(Load_Set_Flags_Com)
;  M 0308  0     ,(Load_Set_Event_Com)
;  M 0309  0     ,(Load_Select_Com)
;  M 0310  0     ,(Load_Start_Com)
;  M 0311  0     ,(Diagnostic_Running)
;  M 0312  0     ,(Dummy_10)
;  M 0313  0     ,(Advance_Diagnostic)
;  M 0314  0     ,(Done_Diagnostics)
;  M 0315  0     ,(Exit_CRD)
;  M 0316  0     ,(Dummy_12)
;  M 0317  0     ,(AutoSizer_Error)
;  M 0318  0     ,(Internal_Error)
;  M 0319  0     ,(Load_Error)
;  M 0320  0     ,(Start_Error)
;  M 0321  0     ,(Diagnostic_Error)
;  M 0322  0     ,(Preparation_Error)
;  M 0323  0     ,(Dummy_13)
;  M 0324  0     ,(Last_State)
;  M 0325  0       ! This constant is the last such number in this list. It must remain
;  M 0326  0       ! in this position (i.e., last). If any states are added to this list,
;  M 0327  0       ! they must be added before this one.
;  M 0328  0    );
;  M 0329  0
;  M 0330  0   LITERAL
;  M 0331  0    $EquLst (MM$K_State_, GLOBAL, 0, 1
;  M 0332  0       ! These constants define the values used to access the columns in the
;  M 0333  0       ! CRD$GAW_MM_State_Table. These are the various states that CRD Menu Test can
;  M 0334  0       ! be in for the Main Menu.
;  M 0335  0      ,(Dummy_1)
;  M 0336  0      ,(Dummy_2)
;  M 0337  0      ,(Dummy_3)
;  M 0338  0     ,(Start)
;  M 0339  0      ,(Dummy_5)
;  M 0340  0     ,(Print_Menu_Heading)
;  M 0341  0     ,(Load_AutoSizer)
;  M 0342  0     ,(Set_Flags_AutoSizer)
;  M 0343  0     ,(Start_AutoSizer)
;  M 0344  0     ,(AutoSizer_Running)
;  M 0345  0     ,(Build_DDBs)
;  M 0346  0     ,(Build_HSTs)
;  M 0347  0      ,(Dummy_7)
;  M 0348  0     ,(Done_Inits)
```

```
; M 0349  0      ,(Dummy_8)
; M 0350  0     ,(Print_Menu)
; M 0351  0     ,(Change_Table)
; M 0352  0     ,(Menu_Prompt)
; M 0353  0     ,(Internal_Error)
; M 0354  0     ,(AutoSizer_Error)
; M 0355  0     ,(Last_State)
; M 0356  0       ! This constant is the last such number in this list. It must remain
; M 0357  0       ! in this position (i.e., last). If any states are added to this list,
; M 0358  0       ! they must be added before this one.
; M 0359  0    );
; M 0360  0
; M 0361  0   LITERAL
; M 0362  0    $EquLst (TQ$K_State_, GLOBAL, 0, 1
; M 0363  0       ! These constants define the values used to access the columns in the
; M 0364  0       ! CRD$GAW_TQ_State_Table. These are the various states that CRD Menu Test can
; M 0365  0       ! be in for the Test Queue.
; M 0366  0      ,(Dummy_1)
; M 0367  0      ,(Dummy_2)
; M 0368  0      ,(Dummy_3)
; M 0369  0     ,(Init_TQ)
; M 0370  0     ,(Get_DDB)
; M 0371  0     ,(Change_Table)
; M 0372  0      ,(Dummy_4)
; M 0373  0     ,(Done_Test_Queue)
; M 0374  0      ,(Dummy_5)
; M 0375  0     ,(Internal_Error)
; M 0376  0     ,(Last_State)
; M 0377  0       ! This constant is the last such number in this list. It must remain
; M 0378  0       ! in this position (i.e., last). If any states are added to this list,
; M 0379  0       ! they must be added before this one.
; M 0380  0    );
; M 0381  0
; M 0382  0   LITERAL
; M 0383  0    $EquLst (HS$K_State_, GLOBAL, 0, 1
; M 0384  0       ! These constants define the values used to access the columns in the
; M 0385  0       ! CRD$GAW_HS_State_Table. These are the various states that CRD Menu Test can
; M 0386  0       ! be in for the Hardware Support.
; M 0387  0      ,(Dummy_1)
; M 0388  0      ,(Dummy_2)
; M 0389  0      ,(Dummy_3)
; M 0390  0     ,(Init_HS)
; M 0391  0     ,(Advance_Diagnostic)
; M 0392  0     ,(Load_General_Com)
; M 0393  0     ,(Advance_General_Com)
; M 0394  0     ,(Done_General_Coms)
; M 0395  0      ,(Dummy_4)
; M 0396  0     ,(Load_Load_Com)
; M 0397  0     ,(Load_Set_Flags_Com)
; M 0398  0     ,(Load_Set_Event_Com)
; M 0399  0     ,(Load_Select_Com)
; M 0400  0     ,(Load_Start_Com)
; M 0401  0     ,(Diagnostic_Running)
; M 0402  0      ,(Dummy_6)
; M 0403  0     ,(Change Tab e)
```

```
;  M 0404  0     ,(Load_Error)
;  M 0405  0     ,(Start_Error)
;  M 0406  0     ,(Diagnostic_Error)
;  M 0407  0     ,(Preparation_Error)
;  M 0408  0     ,(Internal_Error)
;  M 0409  0     ,(Last_State)
;  M 0410  0       ! This constant is the last such number in this list. It must remain
;  M 0411  0       ! in this position (i.e., last). If any states are added to this list,
;  M 0412  0       ! they must be added before this one.
;  M 0413  0     );
;  M 0414  0
;  M 0415  0   LITERAL
;  M 0416  0    $EquLst (CRD$K_, GLOBAL, 0, 1
;  M 0417  0       ! These constants define the types of messages that CRD will output.
;  M 0418  0       ! These are used with the $CRD_Message macro.
;  M 0419  0     ,(Star_Line) ! This message prints asterisks on both sides of the formatted FAO string.
;  M 0420  0     ,(New_Line) ! This message prints a <CR><LF> before the formatted FAO string.
;  M 0421  0     ,(Same_Line) ! This message prints the formatted FAO string only.
;  M 0422  0     ,(Last_Type) ! This constant denotes the last type of message that CRD will output.
;  M 0423  0     );
;  M 0424  0
;  M 0425  0   LITERAL
;  M 0426  0    $EquLst (CRD$K_IDC_, GLOBAL, 0, 1
;  M 0427  0       ! These constants define the names of diagnostics that CRD-AutoTest will run
;  M 0428  0       ! for IDC based system       [01]
;  M 0429  0     ,(EVRAD_0) ! Non-destructive disk tester for the R80
;  M 0430  0       ! or RL02, DQA0
;  M 0431  0     ,(EVRAD_1) ! Non-destructive disk tester for DQA1
;  M 0432  0     ,(EVDLB) ! Combo Synchronous port tester
;  M 0433  0     ,(EVDLC) ! Combo Asynchronous port tester
;  M 0434  0     ,(EVDLD) ! Combo Parallel port tester
;  M 0435  0     ,(Last_Diagnostic)
;  M 0436  0       ! The last diagnostic. This must always be the last such constant in this list.
;  M 0437  0     );
;  M 0438  0
;  M 0439  0
;  M 0440  0   LITERAL     !         [01]
;  M 0441  0    $EquLst (CRD$K_LESI_, GLOBAL, 0, 1
;  M 0442  0       ! These constants define the names
;  M 0443  0       ! of diagnostics that CRD-AutoTest will run
;  M 0444  0       ! for AZTEC LESI based systems
;  M 0445  0     ,(EVRMB_0) ! Non-destructive disk tester for the RC25
;  M 0446  0       ! Drive 0
;  M 0447  0     ,(EVRMB_1) ! Non-destructive disk tester for RC25
;  M 0448  0       ! Drive 1
;  M 0449  0     ,(ENWCA) ! Non-destructive console/graphic PEARL    ![03]
;  M 0450  0       ! level 3 diagnostic
;  M 0451  0     ,(Last_Diagnostic)
;  M 0452  0       ! The last diagnostic. This must always be the last such constant in this list.
;  M 0453  0     );
;  M 0454  0
;  M 0455  0
;  M 0456  0
;  M 0457  0   LITERAL     !         [01]
;  M 0458  0    $EquLst (CRD$K_UDA_3_, GLOBAL, 0, 1
```

```
; M 0459  0      ! These constants define the names
; M 0460  0      ! of diagnostics that CRD-AutoTest will run
; M 0461  0      ! for a UDA based system with
; M 0462  0      ! an RA60 for Drive 1 and RA60 Drive 0.
; M 0463  0    ,(EVRLA_0) ! Non-destructive disk tester for
; M 0464  0      ! Drive 0
; M 0465  0    ,(EVRLA_1) ! Non-destructive disk tester for
; M 0466  0      ! Drive 1
; M 0467  0    ,(EVDLB) ! Combo Synchronous port tester
; M 0468  0    ,(EVDLC) ! Combo Asynchronous port tester
; M 0469  0    ,(EVDLD) ! Combo Parallel port tester
; M 0470  0    ,(Last_Diagnostic)
; M 0471  0      ! The last diagnostic. This must always be the last such constant in this list.
; M 0472  0   );
; M 0473  0
; M 0474  0
; M 0475  0  LITERAL    !         [01]
; M 0476  0   $EquLst (CRD$K_UDA_2_, GLOBAL, 0, 1
; M 0477  0      ! These constants define the names
; M 0478  0      ! of diagnostics that CRD-AutoTest will run
; M 0479  0      ! for a UDA based system with
; M 0480  0      ! no RA60 for Drive 1 and an RA60 Drive 0.
; M 0481  0    ,(EVRLA_0) ! Non-destructive disk tester for
; M 0482  0      ! Drive 0
; M 0483  0    ,(EVDLB) ! Combo Synchronous port tester
; M 0484  0    ,(EVDLC) ! Combo Asynchronous port tester
; M 0485  0    ,(EVDLD) ! Combo Parallel port tester
; M 0486  0    ,(Last_Diagnostic)
; M 0487  0      ! The last diagnostic. This must always be the last such constant in this list.
; M 0488  0   );
; M 0489  0
; M 0490  0  LITERAL    !         [01]
; M 0491  0   $EquLst (CRD$K_UDA_1_, GLOBAL, 0, 1
; M 0492  0      ! These constants define the names
; M 0493  0      ! of diagnostics that CRD-AutoTest will run
; M 0494  0      ! for a UDA based system with
; M 0495  0      ! an RA60 for Drive 1 and RA80 or 81 Drive 0.
; M 0496  0    ,(EVRLA_0) ! Non-destructive disk tester for
; M 0497  0      ! Drive 0
; M 0498  0    ,(EVRLA_1) ! Non-destructive disk tester for
; M 0499  0      ! Drive 1
; M 0500  0    ,(EVDLB) ! Combo Synchronous port tester
; M 0501  0    ,(EVDLC) ! Combo Asynchronous port tester
; M 0502  0    ,(EVDLD) ! Combo Parallel port tester
; M 0503  0    ,(Last_Diagnostic)
; M 0504  0      ! The last diagnostic. This must always be the last such constant in this list.
; M 0505  0   );
; M 0506  0
; M 0507  0
; M 0508  0  LITERAL    !         [01]
; M 0509  0   $EquLst (CRD$K_UDA_0_, GLOBAL, 0, 1
; M 0510  0      ! These constants define the names
; M 0511  0      ! of diagnostics that CRD-AutoTest will run
; M 0512  0      ! for a UDA based system with
; M 0513  0      ! no RA60 for Drive 1 and an RA80 or 81 for Drive 0.
```

```
; M 0514  0     ,(EVRLA_0) ! Non-destructive disk tester for
; M 0515  0       ! Drive 0
; M 0516  0     ,(EVDLB) ! Combo Synchronous port tester
; M 0517  0     ,(EVDLC) ! Combo Asynchronous port tester
; M 0518  0     ,(EVDLD) ! Combo Parallel port tester
; M 0519  0     ,(Last_Diagnostic)
; M 0520  0       ! The last diagnostic. This must always be the last such constant in this list.
; M 0521  0    );
; M 0522  0
; M 0523  0   LITERAL
; M 0524  0    $EquLst (CRD$K_, GLOBAL, 0, 1
; M 0525  0       ! These constants define the column numbers for the Hardware Support table.
; M 0526  0     ,(Diagnostic_Name)
; M 0527  0       ! A pointer to an ASCIC string
; M 0528  0     ,(Set_Flags_Args)
; M 0529  0       ! A pointer to an ASCIC string
; M 0530  0     ,(Set_Event_Args)
; M 0531  0       ! A pointer to an ASCIC string
; M 0532  0     ,(Device_Name)
; M 0533  0       ! A pointer to an ASCIC string in the PTable
; M 0534  0     ,(Start_Qualifiers)
; M 0535  0       ! A pointer to an ASCIC string
; M 0536  0     ,(Test_Start_Text)
; M 0537  0       ! A pointer to an ASCIC string
; M 0538  0     ,(RunTime)
; M 0539  0       ! A pointer to an ASCIC string
; M 0540  0     ,(Preparation_Error_Text)
; M 0541  0       ! A pointer to an ASCIC string
; M 0542  0     ,(Last_Entry)
; M 0543  0       ! The last column number (+ 1) in this list. This must always be last
; M 0544  0       ! in this list.
; M 0545  0    );
; M 0546  0
; M 0547  0   LITERAL
; M 0548  0    $EquLst (CRD$K_DDB_, GLOBAL, 0, 1
; M 0549  0       ! These constants define the entries in a DDB
; M 0550  0     ,(FLink)
; M 0551  0       ! Forward link
; M 0552  0     ,(BLink)
; M 0553  0       ! Backwards link
; M 0554  0     ,(PTable_Entry)
; M 0555  0       ! Pointer to a PTable entry
; M 0556  0     ,(Auto_Support_Entry)
; M 0557  0         ! Pointer to a row in the Hardware_Support_Table (for CRD Auto Test)
; M 0558  0     ,(Menu_Support_Entry)
; M 0559  0       ! Pointer to the concatenated Hardware Support Tables (for CRD Menu Test)
; M 0560  0     ,(Status)
; M 0561  0       ! Three bytes containing status bits and a byte contaning the size
; M 0562  0       ! of the allocated DDB - in BYTEs
; M 0563  0     ,(Numb_Support_Tables)
; M 0564  0       ! Number of concatenated Hardware
; M 0565  0       ! Support Blocks pointed to by DDB's
; M 0566  0       ! Man_Support_Entry
; M 0567  0     ,(Menu_Device_Numb)
; M 0568  0       ! The select for test number specified
```

```
;  M 0569   0      ! for the DDB's device.
;  M 0570   0    ,(SupBlk_Block_Ptr)
;  M 0571   0      ! Pointer to the device support
;  M 0572   0      ! block located in the loaded CRD MENU
;  M 0573   0      ! Support file database .
;  M 0574   0    ,(Memory_Size) ! Size in bytes of allocated
;  M 0575   0      ! memory space for DDB
;  M 0576   0    ,(Menu_Support_Size)
;  M 0577   0      ! Size in bytes of allocated memory
;  M 0578   0      ! space for 1 or more concatenated
;  M 0579   0      ! hardware support tables pointed to
;  M 0580   0      ! by [CRD$K_DDB_Menu_Support_Entry].
;  M 0581   0    ,(Last_Entry)
;  M 0582   0      ! This is the last DDB entry. It must ALWAYS be last.
;  M 0583   0   );
;  M 0584   0
;  M 0585   0  LITERAL
;  M 0586   0   $EquLst (AUTO$K_Exit_, GLOBAL, 0, 1
;  M 0587   0      ! These literals define the reasons for stopping CRD-AutoTest.
;  M 0588   0    ,(Control_C)
;  M 0589   0      ! Exit CRD because a Control-C was typed.
;  M 0590   0    ,(Internal_Error)
;  M 0591   0      ! Stop CRD because of an internal error (i.e., software bug).
;  M 0592   0    ,(AutoSizer_Error)
;  M 0593   0      ! Exit CRD because the AutoSizer has a problem.
;  M 0594   0    ,(Errors_Complete)
;  M 0595   0    ,(Errors_Incomplete)
;  M 0596   0    ,(NoErrors_Complete)
;  M 0597   0    ,(NoErrors_Incomplete)
;  M 0598   0    ,(NoErrors_Prep)
;  M 0599   0    ,(Inv_Base_System)
;  M 0600   0      ! Invalid Base System.  CRD unable
;  M 0601   0      ! to determine Base System Type for test.    [01]
;  M 0602   0    ,(Dummy_2) ! Use one of these two when more reasons are added to this list.
;  M 0603   0      ! This must be done so that the next set of literals come out to be
;  M 0604   0    ,(Dummy_3) ! correct - note they use the value of AUTO$K_Exit_Last_Reason.
;  M 0605   0    ,(Last_Reason)
;  M 0606   0      ! The last stop reason. This must ALWAYS be the last stop reason in this
;  M 0607   0      ! list. Any additions to this list must be made before this name.
;  M 0608   0   );
;  M 0609   0
;  M 0610   0  LITERAL
;  M 0611   0   $EquLst (MENU$K_Exit_, GLOBAL, AUTO$K_Exit_Last_Reason, 1
;  M 0612   0      ! These literals define the reasons for stopping CRD Menu Test.
;  M 0613   0    ,(Operator_Stop)
;  M 0614   0      ! Exit CRD because the operator requested an exit from the Main Menu.
;  M 0615   0    ,(Operator_Abort)
;  M 0616   0      ! Exit CRD because the operator requested an exit from the ^C menu.
;  M 0617   0    ,(Internal_Error)
;  M 0618   0      ! Stop CRD Menu Test because of an internal error (i.e., software bug).
;  M 0619   0    ,(AutoSizer_Error)
;  M 0620   0      ! Exit CRD Menu Test because the AutoSizer has a problem.
;  M 0621   0    ,(Sup_File_Not_Found)
;  M 0622   0    ,(Sup_File_Load_Err)
;  M 0623   0    ,(Last_Reason)
```

```
; M 0624  0      ! The last stop reason. This must ALWAYS be the last stop reason in this
; M 0625  0      ! list. Any additions to this list must be made before this name.
; M 0626  0    );
; M 0627  0
; M 0628  0  LITERAL
; M 0629  0    $EquLst (CRD$K_, GLOBAL, -1, -1
; M 0630  0      ! These literals define the internal error codes. They start at -1 and descend
; M 0631  0      ! from there. The reason for this is that they must be distinguishable from
; M 0632  0      ! the TypeCodes.
; M 0633  0     ,(Allocation_Error)
; M 0634  0      ! An error occurred in allocating memory for the DDB's.
; M 0635  0     ,(Action_Range_Error)
; M 0636  0      ! The action routine number is out of range.
; M 0637  0     ,(Action_EQL_Do_Nothing)
; M 0638  0      ! The action routine number eql Do_Nothing.
; M 0639  0     ,(Bad_TypeCode_Error)
; M 0640  0      ! The typecode is ivalid in the current context.
; M 0641  0     ,(Data_Length_Error)
; M 0642  0      ! The length of the buffer to insert the general commands into is not
; M 0643  0      ! long enough to hold a general command.
; M 0644  0     ,(MM_Internal_Error)
; M 0645  0     ,(TQ_Internal_Error)
; M 0646  0     ,(HS_Internal_Error)
; M 0647  0     ,(Bad_Action_Number)
; M 0648  0     ,(Load_Sup_File)
; M 0649  0     ,(Create_HST)
; M 0650  0     ,(Last_Internal_Error)
; M 0651  0      ! This is the last literal for the Internal Error codes.
; M 0652  0    )
;   0653  0  %;
;   0654  0
; M 0655  0 MACRO $SCREEN_Literals =
; M 0656  0  LITERAL
; M 0657  0    $EquLst (SCREEN$K_, GLOBAL, 0, 1
; M 0658  0      ! These constants determine which soft connsole
; M 0659  0      ! operation to perform.
; M 0660  0     ,(Press_Return)
; M 0661  0     ,(Clear_Screen)
; M 0662  0     ,(Count_Line)
; M 0663  0     ,(Press_Return_MM)
; M 0664  0     ,(Press_Return_TM)
; M 0665  0     ,(TM_Msg)
; M 0666  0    )
;   0667  0  %;
```

```
;    0668  0 %SBTTL 'Other CRD Literals'
;    0669  0
;    0670  0 LITERAL
;    0671  0  Off        = 0,
;    0672  0  On         = 1,
;    0673  0
;    0674  0  False        = 0,
;    0675  0  True         = 1,
;    0676  0
;    0677  0  CRD$K_Failure = 0,
;    0678  0  CRD$K_Success = 1,
;    0679  0
;    0680  0  CRD$K_Return_Failure = 0,
;    0681  0  CRD$K_Return_Success = 1,
;    0682  0  CRD$K_Repeat_Turing  = 2,
;    0683  0
;    0684  0
;    0685  0  CRD$K_Numb_Screen_Lines = 22,
;    0686  0      ! Maximum number of lines on soft
;    0687  0      ! console screen for output which
;    0688  0      ! allows space for a msg on line 24.
;    0689  0
;    0690  0  CRD$K_DS_Command_Size     = 80,
;    0691  0      ! Maximum size of each general command
;    0692  0  CRD$K_Numb_General_Commands = 3,
;    0693  0      ! The number of general commands
;    0694  0  CRD$K_IDC_Numb_Diagnostics    = 5,
;    0695  0      ! The number of diagnostics used in CRD-AutoTest
;    0696  0      ! for an IDC based system      [01]
;    0697  0  CRD$K_LESI_Numb_Diagnostics    = 3,
;    0698  0      ! The number of diagnostics used in CRD-AutoTest    ![03]
;    0699  0      ! for an AZTEC LESI based system     [01]
;    0700  0  CRD$K_UDA_0_Numb_Diagnostics    = 4,
;    0701  0      ! The number of diagnostics used in CRD-AutoTest
;    0702  0      ! for a UDA based system with
;    0703  0      ! no RA60 for Drive 1 and RA80 or 81 for Drive 0.   [01]
;    0704  0  CRD$K_UDA_1_Numb_Diagnostics    = 5,
;    0705  0      ! The number of diagnostics used in CRD-AutoTest
;    0706  0      ! for a UDA50 based system     [01]
;    0707  0      ! with a RA60 for Drive 1 and RA80 or 81 Drive 0.   [01]
;    0708  0  CRD$K_UDA_2_Numb_Diagnostics    = 4,
;    0709  0      ! The number of diagnostics used in CRD-AutoTest
;    0710  0      ! for a UDA based system with
;    0711  0      ! no RA60 for Drive 1 and RA80 or 81 for Drive 0.   [01]
;    0712  0  CRD$K_UDA_3_Numb_Diagnostics    = 5,
;    0713  0      ! The number of diagnostics used in CRD-AutoTest
;    0714  0      ! for a UDA50 based system     [01]
;    0715  0      ! with a RA60 for Drive 1 and RA80 or 81 Drive 0.   [01]
;    0716  0  CRD$K_Numb_Support_Entries  = 8,
;    0717  0      ! The number of entries per row in the Hardware_Support table
;    0718  0  CRD$K_Numb_Dispatch_Vectors = 14,
;    0719  0      ! This MUST be changed whenever the number of vectors is changed in
;    0720  0      ! CRDVECS or DSVECS. This is used by the CRDImage module to allocate
;    0721  0      ! space to store the original DS dispatch vectors (in DSVECS). It is also
;    0722  0      ! used to allocate static space for the debug vector table (below).
```

```
;    0723  0      ! Note that this is not the actual number of vectors in the CRDVECS (DSVECS)
;    0724  0      ! module. It is the number of vectors that are currently in use.
;    0725  0  CRD$K_Numb_UnUsed_Vectors   = 11,
;    0726  0      ! This is the number of vectors that are unused in the vector tables in
;    0727  0      ! DSVECS and CRDVECS. This number plus the number above MUST equal the
;    0728  0      ! CRD$K_Total_Numb_Vectors constant.
;    0729  0  CRD$K_Total_Numb_Vectors    = 25,
;    0730  0      ! The total number of used and unused vectors in the DSVECS and CRDVECS
;    0731  0      ! modules. CRD$K_Total_Numb_Vectors = CRD$K_Numb_UnUsed_Vectors +
;    0732  0      ! CRD$K_Numb_Dispatch_Vectors. ALWAYS!
;    0733  0
;    0734  0  CRD$K_Numb_States    = 36, ! The number of states in the CRD$GW_State_Table.
;    0735  0  MM$K_Numb_States     = 20, ! The number of states in the CRD$GAW_MM_State_Table.
;    0736  0  TQ$K_Numb_States     = 10, ! The number of states in the CRD$GAW_TQ_State_Table.
;    0737  0  HS$K_Numb_States     = 22, ! The number of states in the CRD$GAW_HS_State_Table.
;    0738  0
;    0739  0  CRD$K_Numb_TypeCodes = 38, ! The number of defined typecodes.
;    0740  0
;    0741  0
;    0742  0  CRD$K_Action_Routine = 0, ! These two literals are used to access either the high word or the low
;    0743  0  CRD$K_New_State      = 1, ! word of each entry in the CRD$GW_State_Table (New_State = high word).
;    0744  0
;    0745  0
;    0746  0  CRD$K_CRD_Message_Buffer_Len = 132,
;    0747  0      ! The number of bytes in the message buffer.
;    0748  0
;    0749  0
;    0750  0  CRD$K_DDB_Size = 44;  ! The size of a Device Data Block = 44 bytes (11 longwords). Note that this
;    0751  0      ! is the size that is needed for both Auto Test and Menu Test.
;    0752  0
;    0753  0
;    0754  0
```

```
;    0755  0 %SBTTL 'DDB Literals and Macros'
;    0756  0
;    0757  0  !+
;    0758  0  ! Use fields to get to the specific bits in the DDB Status longword.
;    0759  0  !-
;    0760  0
;    0761  0 MACRO DDB$V_Status_Diagnostic_Error = 0, 1, 0 %;
;    0762  0       ! The test for this device encountered a diagnostic error
;    0763  0
;    0764  0 MACRO DDB$V_Status_Preparation_Error = 1, 1, 0 %;
;    0765  0       ! The test for this device encountered a preparation error
;    0766  0
;    0767  0 MACRO DDB$V_Status_Essential_Device = 2, 1, 0 %;
;    0768  0       ! This device must be tested. If it is not, then in AutoTest, the run
;    0769  0       ! is considered INCOMPLETE.
;    0770  0
;    0771  0 MACRO DDB$V_Status_Check_Device_Type = 3, 1, 0 %;
;    0772  0       ! Replace the device type field in the Test_Start_Text when
;    0773  0       ! it is being output. Replace character for character with
;    0774  0       ! the Device_Type ascic string in the PTable.
;    0775  0
;    0776  0 MACRO DDB$V_Status_Device_Bad  = 4, 1, 0 %;
;    0777  0       ! If one diagnostic in a set of HSB's finds an error (thereby setting
;    0778  0       ! Diagnostic_Error - above - to True), this will be True also.
;    0779  0
;    0780  0 MACRO DDB$V_Status_Dev_Test_Complete = 5, 1, 0 %;
;    0781  0       ! If true, all the diagnostics for a specific DDB completed
;    0782  0       ! successfully.
;    0783  0
;    0784  0 MACRO DDB$V_Status_Message_Printed = 6, 1, 0 %;
;    0785  0       ! If true, a message was printed by an error routine. If not true, then no
;    0786  0       ! message has been printed for this diagnostic, and we must print the PASS
;    0787  0       ! message.
;    0788  0
;    0789  0 MACRO
;    0790  0  DDB$B_Status_DDB_Size  = 24, 8, 0 %;
;    0791  0       ! The size in BYTEs of the allocated DDB.
```

```
;    0792  0 %SBTTL 'CRD_Logic_Error Macro'
;    0793  0
;    0794  0 MACRO
;    0795  0  !+
;    0796  0  ! Usage:
;    0797  0  ! CRD_Logic_Error ('Error message to be printed out');
;    0798  0  ! Function:
;    0799  0  ! This macro outputs an error message that will indicate that something is wrong with CRD.
;    0800  0  !-
;    0801  0
; M  0802  0  CRD_Logic_Error (Error_String) =
; M  0803  0   (
; M  0804  0   BIND
; M  0805  0    ES = $ASCIC ('!/** CRD Internal Logic Error ** ', Error_String, '.!/');
; M  0806  0
; M  0807  0   $CRD_Print (ES)
; M  0808  0   )
;    0809  0  %;
```

```
;    0810  0 %SBTTL 'Logic_Error Macro'
;    0811  0
;    0812  0 MACRO
;    0813  0  !+
;    0814  0  ! Usage:
;    0815  0  ! Logic_Error ('Error message to be printed out');
;    0816  0  ! Function:
;    0817  0  ! This macro outputs an error message that will indicate that something is wrong with CRD.
;    0818  0  ! This macro can only be used in the Resident-DS image.
;    0819  0  !-
;    0820  0
; M  0821  0  Logic_Error (Error_String) =
; M  0822  0   (
; M  0823  0   $DS_TypeDef;  ! Define the typcodes locally
; M  0824  0
; M  0825  0   BIND
; M  0826  0    ES = $ASCIC ('!/** CRD(R) Internal Logic Error ** ', Error_String, '.!/');
; M  0827  0
; M  0828  0   $Print (DS$K_Type_CRD_AutoTest, DS$K_PrintF, ES)
; M  0829  0   )
;    0830  0  %;
```

```
;    0831  0 %SBTTL 'Assert Macro'
;    0832  0
;    0833  0 MACRO
;    0834  0  !+
;    0835  0  ! Usage:
;    0836  0  ! CRD_ASSERT ((.A GTR .B), 'A <= B');
;    0837  0  !
;    0838  0  ! If the specified condition is not true, an error is printed by the Bliss compiler. If it is true,
;    0839  0  ! that is stated.
;    0840  0  !-
;    0841  0
; M  0842  0  CRD_Assert (Condition, Assert_String) =
; M  0843  0   %IF %CTCE (Condition)
; M  0844  0   %THEN
; M  0845  0    %IF (Condition)
; M  0846  0    %THEN
; M  0847  0     %PRINT ('ASSERT: The assertion is true.')
; M  0848  0    %ELSE
; M  0849  0     %IF %ISSTRING (Assert_String)
; M  0850  0     %THEN
; M  0851  0      %ERROR ('ASSERT: The assertion is not true:')
; M  0852  0      %ERROR ('ASSERT:       ', Assert_String, '!')
; M  0853  0     %ELSE
; M  0854  0      %ERROR ('ASSERT: The assertion is not true!')
; M  0855  0     %FI
; M  0856  0    %FI
; M  0857  0   %ELSE
; M  0858  0    %ERRORMACRO ('ASSERT: The assertion is not a compiletime constant expression!')
; M  0859  0   %FI
;    0860  0  %;
;    0861  0
```

```
;    0862  0 %SBTTL 'Debug Macro'
;    0863  0
;    0864  0 MACRO
;    0865  0  !+
;    0866  0  ! This macro is used only in the debugging stages. It's only parameter is a string giving, e.g., the
;    0867  0  ! name of a routine. DEBUG is turned on whenever the compilation is made with the /VARIANT:n qual'f'er,
;    0868  0  ! where n <> 0.
;    0869  0  !-
;    0870  0
; M  0871  0  CRD_DEBUG (Debug_String) =
; M  0872  0   %IF %VARIANT
; M  0873  0   %THEN
; M  0874  0    (
; M  0875  0    BIND
; M  0876  0     DS = $ASCIC ('!/** CRD-Debug ** ', Debug_Str'ng, '.!/');
; M  0877  0
; M  0878  0    $CRD_Print (DS)
; M  0879  0    )
; M  0880  0   %ELSE
; M  0881  0    ! No code put in.
; M  0882  0   %FI
;    0883  0  %;
```

```
;    0884  0 %SBTTL 'Boolean Macro'
;    0885  0
;    0886  0 MACRO
;    0887  0   !+
;    0888  0   ! Define a macro that takes an expression as an argument. The value of this macro is either
;    0889  0   ! one or zero. All bits but the least significant one are cleared.
;    0890  0   !-
;    0891  0
; M  0892  0   Boolean (Expression) =
; M  0893  0    ((Expression) AND 1)
;    0894  0   %;
```

```
;   0895  0 %SBTTL '$CRD_Print Macro'
;   0896  0
;   0897  0 MACRO
; M 0898  0  $CRD_Print (Format) =
; M 0899  0   (
; M 0900  0   $DS_TypeDef;  ! Define the typecodes locally
; M 0901  0
; M 0902  0   (.CRD$Print)
; M 0903  0    (
; M 0904  0    DS$K_Type_CRD_AutoTest,
; M 0905  0    DS$K_PrintF,
; M 0906  0    Format
; M 0907  0    %IF %LENGTH GTR 1
; M 0908  0    %THEN
; M 0909  0      , %REMAINING
; M 0910  0    %FI
; M 0911  0    )
; M 0912  0   )
;   0913  0  %;
```

```
;    0914  0 %SBTTL 'Length Macro (for $CRD_Message)'
;    0915  0
;    0916  0 MACRO
; M  0917  0  $CRD_Length [Ascic_String] =
; M  0918  0   (
; M  0919  0   BIND
; M  0920  0    String = (Ascic_String) : VECTOR [, BYTE];
; M  0921  0
; M  0922  0   .String [0]
; M  0923  0   )
;    0924  0  %;
```

```
;    0925  0 %SBTTL 'Cat Macro (for $CRD_Message)'
;    0926  0
;    0927  0 MACRO
; M  0928  0  $CRD_Cat [Ascic_String] =
; M  0929  0   (
; M  0930  0   BIND
; M  0931  0    String = (Ascic_String) : VECTOR [, BYTE];
; M  0932  0
; M  0933  0   .String [0]
; M  0934  0   ), CH$Ptr (Ascic_String + 1)
;    0935  0  %;
;    0936  0
;    0937  0
;    0938  0
;    0939  0 %SBTTL 'Concat Macro (for $CRD_Message)'
;    0940  0
;    0941  0 MACRO
; M  0942  0  $CRD_Concat (Buffer_Address, Buffer_Length) [] =
; M  0943  0   (
; M  0944  0   BIND
; M  0945  0    Address = (Buffer_Address),
; M  0946  0    Length  = (Buffer_Length);
; M  0947  0
; M  0948  0   CH$Fill (%C' ', Length, CH$Ptr (Address));
; M  0949  0    (
; M  0950  0    MAP
; M  0951  0     Address : VECTOR [, BYTE];
; M  0952  0
; M  0953  0    Address [0] = 0 + $CRD_Length (%REMAINING)
; M  0954  0    );
; M  0955  0   CH$Copy ($CRD_Cat (%REMAINING), %C' ', Length - 1, CH$Ptr (Address + 1))
; M  0956  0   )
;    0957  0  %;
```

```
;    0958  0 %SBTTL 'The Message Outputter'
;    0959  0
;    0960  0 MACRO
; M  0961  0  $CRD_Message (Message_Type, Format) [] =
; M  0962  0    !+
; M  0963  0    ! The message_type defines what type of message this is. The types are as follows:
; M  0964  0    ! Same_Line, New_Line, and Star_Line.
; M  0965  0    ! The Format is a ASCIC FAO string.
; M  0966  0    ! The [] are any variables needed to print the string.
; M  0967  0    !-
; M  0968  0
; M  0969  0    (
; M  0970  0    !+
; M  0971  0    ! First print the type of message. If the type of message is  Star_Line , then print the
; M  0972  0    ! message prefix (i.e., it has a prefix and a suffix).
; M  0973  0    !-
; M  0974  0
; M  0975  0    %IF %IDENTICAL (Message_Type, 'Star_Line')
; M  0976  0    %THEN
; M  0977  0     $CRD_Print (CRD$GT_Star_Line_Prefix_Message);
; M  0978  0    %ELSE
; M  0979  0     $CRD_Print (%NAME ('CRD$GT_', Message_Type, '_Message'));
; M  0980  0    %FI
; M  0981  0
; M  0982  0    !+
; M  0983  0    ! Then print the formatted FAO string, along with any values.
; M  0984  0    !-
; M  0985  0
; M  0986  0    $CRD_Print
; M  0987  0     (
; M  0988  0     Format
; M  0989  0     %IF NOT %NULL (%REMAINING)
; M  0990  0     %THEN
; M  0991  0       , %REMAINING
; M  0992  0     %FI
; M  0993  0     )
; M  0994  0
; M  0995  0    !+
; M  0996  0    ! Finally, if it was a Star_Line message, print the suffix.
; M  0997  0    !-
; M  0998  0
; M  0999  0    %IF %IDENTICAL (Message_Type, 'Star_Line')
; M  1000  0    %THEN
; M  1001  0     ; $CRD_Print (CRD$GT_Star_Line_Suffix_Message)
; M  1002  0    %FI
; M  1003  0    )
;    1004  0  %;
```

```
;   1005  0 %SBTTL '$CRD_Return_Length_Status Macro'
;   1006  0
;   1007  0 MACRO
; M 1008  0  $CRD_Return_Length_Status (Status) =
; M 1009  0   RETURN (
; M 1010  0    (CRD$K_DS_Command_Size ^ 16) +
; M 1011  0     %IF %NULL (Status)
; M 1012  0     %THEN
; M 1013  0      CRD$K_Success
; M 1014  0     %ELSE
; M 1015  0      Status
; M 1016  0     %FI
; M 1017  0    )
;   1018  0  %;
;   1019  0
;   1020  0
;   1021  0
```

```
;    1022  0 %SBTTL '$CRD_Queue_Head Macro'
;    1023  0
;    1024  0 FIELD
;    1025  0  Queue_Head_Fields =
;    1026  0   SET
;    1027  0   CRD$V_First_Entry = [0, 0, 32, 0],
;    1028  0   CRD$V_Last_Entry  = [1, 0, 32, 0]
;    1029  0   TES;
;    1030  0
; M  1031  0 MACRO $CRD_Queue_Head =
; M  1032  0   VECTOR [2, LONG]
; M  1033  0    FIELD Queue_Head_Fields
;    1034  0  %;
```

```
;    1035  0 %SBTTL 'Device Data Block_Structure Definition'
;    1036  0
;    1037  0 STRUCTURE
;    1038  0  !+
;    1039  0  ! Note that no allocation-formals are given since these Data Blocks will be allocated
;    1040  0  ! dynamically. Merely define the accessing-formals.
;    1041  0  !-
;    1042  0
;    1043  0  Device_Data_Block_Structure
;    1044  0   [Entry] =
;    1045  1    (
;    1046  1    IF ((Entry * 4) GEQ CRD$K_DDB_Size) THEN
;    1047  1     %X'FFFFFFFF'
;    1048  1    ELSE
;    1049  2     (Device_Data_Block_Structure + (Entry * 4))
;    1050  0    );
```

```
;    1051  0 %SBTTL 'General_Command_Structure Definition'
;    1052  0
;    1053  0 STRUCTURE
;    1054  0  General_Command_Structure
;    1055  0   [Com;
;    1056  0    Numb_Coms, Com_Size = CRD$K_DS_Command_Size] =
;    1057  0    [Numb_Coms * Com_Size]
;    1058  1    (
;    1059  1    (General_Command_Structure + (Com * Com_Size));
;    1060  1
;    1061  1    IF (Com GEQ Numb_Coms) THEN
;    1062  1     %X'FFFFFFFF'
;    1063  1    ELSE
;    1064  2     (General_Command_Structure + (Com * Com_Size))
;    1065  0    );
;    1066  0 %SBTTL 'State_Table_Structure Definition'
;    1067  0
;    1068  0 STRUCTURE
;    1069  0  State_Table_Structure
;    1070  0   [TypeCode, State, Contents;
;    1071  0    Numb_TypeCodes, Numb_States] =
;    1072  0    [Numb_TypeCodes * Numb_States * 4]
;    1073  1    (
;    1074  1    (State_Table_Structure + (TypeCode*Numb_States*4) + (State*4));
;    1075  1
;    1076  1    IF (TypeCode GEQ Numb_TypeCodes) THEN
;    1077  1     %X'FFFFFFFF'
;    1078  1    ELSE IF (State GEQ Numb_States) THEN
;    1079  1     %X'FFFFFFFF'
;    1080  1    ELSE
;    1081  2     (State_Table_Structure + (TypeCode*Numb_States*4) + (State*4))
;    1082  0    ) <Contents * 16, 16>;
;    1083  0 %SBTTL 'MM_State_Table_Structure Definition'
;    1084  0
;    1085  0 STRUCTURE
;    1086  0  MM_State_Table_Structure
;    1087  0   [TypeCode, State, Contents;
;    1088  0    Numb_TypeCodes, Numb_States] =
;    1089  0    [Numb_TypeCodes * Numb_States * 4]
;    1090  1    (
;    1091  1    (MM_State_Table_Structure + (TypeCode*Numb_States*4) + (State*4));
;    1092  1
;    1093  1    IF (TypeCode GEQ Numb_TypeCodes) THEN
;    1094  1     %X'FFFFFFFF'
;    1095  1    ELSE IF (State GEQ Numb_States) THEN
;    1096  1     %X'FFFFFFFF'
;    1097  1    ELSE
;    1098  2     (MM_State_Table_Structure + (TypeCode*Numb_States*4) + (State*4))
;    1099  0    ) <Contents * 16, 16>;
;    1100  0 %SBTTL 'TQ_State_Table_Structure Definition'
;    1101  0
;    1102  0 STRUCTURE
;    1103  0  TQ_State_Table_Structure
;    1104  0   [TypeCode, State, Contents;
;    1105  0    Numb_TypeCodes, Numb_States] =
```

```
;    1106  0     [Numb_TypeCodes * Numb_States * 4]
;    1107  1     (
;    1108  1     (TQ_State_Table_Structure + (TypeCode*Numb_States*4) + (State*4));
;    1109  1
;    1110  1     IF (TypeCode GEQ Numb_TypeCodes) THEN
;    1111  1      %X'FFFFFFFF'
;    1112  1     ELSE IF (State GEQ Numb_States) THEN
;    1113  1      %X'FFFFFFFF'
;    1114  1     ELSE
;    1115  2      (TQ_State_Table_Structure + (TypeCode*Numb_States*4) + (State*4))
;    1116  0     ) <Contents * 16, 16>;
;    1117  0 %SBTTL 'HS_State_Table_Structure Definition'
;    1118  0
;    1119  0 STRUCTURE
;    1120  0  HS_State_Table_Structure
;    1121  0   [TypeCode, State, Contents;
;    1122  0    Numb_TypeCodes, Numb_States] =
;    1123  0    [Numb_TypeCodes * Numb_States * 4]
;    1124  1    (
;    1125  1    (HS_State_Table_Structure + (TypeCode*Numb_States*4) + (State*4));
;    1126  1
;    1127  1     IF (TypeCode GEQ Numb_TypeCodes) THEN
;    1128  1      %X'FFFFFFFF'
;    1129  1     ELSE IF (State GEQ Numb_States) THEN
;    1130  1      %X'FFFFFFFF'
;    1131  1     ELSE
;    1132  2      (HS_State_Table_Structure + (TypeCode*Numb_States*4) + (State*4))
;    1133  0     ) <Contents * 16, 16>;
;    1134  0 %SBTTL 'Hardware_Support_Structure Definition'
;    1135  0
;    1136  0 STRUCTURE
;    1137  0  Hardware_Support_Structure
;    1138  0   [Diag, Entry;
;    1139  0    Numb_Diags, Numb_Entries] =
;    1140  0    [Numb_Diags * Numb_Entries * 4]
;    1141  1    (
;    1142  1    (Hardware_Support_Structure + (Diag * Numb_Entries * 4) + (Entry * 4));
;    1143  1
;    1144  1     IF (Diag GEQ Numb_Diags) THEN
;    1145  1      %X'FFFFFFFF'
;    1146  1     ELSE IF (Entry GEQ Numb_Entries) THEN
;    1147  1      %X'FFFFFFFF'
;    1148  1     ELSE
;    1149  2      (Hardware_Support_Structure + (Diag * Numb_Entries * 4) + (Entry * 4))
;    1150  0     );
;    1151  0 %SBTTL 'Hardware_Support_Vector Structure Definition'
;    1152  0
;    1153  0 STRUCTURE
;    1154  0  !+
;    1155  0  ! Define this structure with no allocation-formals. It will be used to MAP storage
;    1156  0  ! to a particular vector in the Hardware_Support_Table.
;    1157  0  !-
;    1158  0
;    1159  0  Hardware_Support_Vector
;    1160  0   [Entry] =
```

```
;   1161  1    (
;   1162  1    IF (Entry GEQ CRD$K_Numb_Support_Entries) THEN
;   1163  1     %X'FFFFFFFF'
;   1164  1    ELSE
;   1165  2     (Hardware_Support_Vector + (Entry * 4))
;   1166  0    );
```

```
;    1167  0 %SBTTL 'Linkages'
;    1168  0
;    1169  0 LINKAGE
;    1170  0  JSB_Begin    = JSB,
;    1171  0
;    1172  0  JSB_Scan_Numeric = JSB (
;    1173  0     REGISTER = 3, REGISTER = 4, REGISTER = 5;
;    1174  0     REGISTER = 0
;    1175  0     ),
;    1176  0
;    1177  0  JSB_NO_REGS = JSB : NOPRESERVE (2,3,4,5,6,7,8,9,10,11),
;    1178  0
;    1179  0  JSB_R0_R1    = JSB (REGISTER = 0, REGISTER = 1) : NOPRESERVE (3,4,5,6,7,8,9,10,11) PRESERVE (2);
```

```
;    1180  0 %SBTTL 'External Routine Declarations'
;    1181  0
;    1182  0 EXTERNAL ROUTINE
;    1183  0  !+
;    1184  0  ! From the CRDACT1 module:
;    1185  0  !-
;    1186  0
;    1187  0  CRD$AutoSizer_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    1188  0      ! The AutoSizer had a problem.
;    1189  0
;    1190  0  CRD$Diagnostic_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    1191  0      ! A diagnostic reported an ERRxxxx.
;    1192  0
;    1193  0  CRD$Exit_CRD   : ADDRESSING_MODE (LONG_RELATIVE),
;    1194  0      ! Exit the CRD process
;    1195  0
;    1196  0  CRD$Internal_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    1197  0      ! An internal error (i.e., ERRSUP or CRD) occurred.
;    1198  0
;    1199  0  CRD$Load_Error   : ADDRESSING_MODE (LONG_RELATIVE),
;    1200  0      ! An error occurred in loading a diagnostic
;    1201  0
;    1202  0  CRD$Preparation_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    1203  0      ! An ERRPREP occurred while running a diagnostic.
;    1204  0
;    1205  0  CRD$Print_DDB   : ADDRESSING_MODE (LONG_RELATIVE),
;    1206  0      ! Print the contents of a DDB
;    1207  0
;    1208  0  CRD$Start_Error   : ADDRESSING_MODE (LONG_RELATIVE),
;    1209  0      ! An error occurred starting a diagnostic
;    1210  0
;    1211  0  !+
;    1212  0  ! From the CRDACT2 module:
;    1213  0  !-
;    1214  0
;    1215  0  CRD$Advance_Diagnostic  : ADDRESSING_MODE (LONG_RELATIVE),
;    1216  0      ! Advance to the next diagnostic
;    1217  0
;    1218  0  CRD$Advance_General_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    1219  0      ! Go to the next general command
;    1220  0
;    1221  0  CRD$Assign_PTable_Ptrs  : ADDRESSING_MODE (LONG_RELATIVE),
;    1222  0      ! Insert the PTable pointers into the DDBs
;    1223  0
;    1224  0  CRD$AutoSizer_Set_Flags  : ADDRESSING_MODE (LONG_RELATIVE),
;    1225  0      ! Load the AutoSizer's SET FLAGS command.
;    1226  0
;    1227  0  CRD$Done_General_Coms  : ADDRESSING_MODE (LONG_RELATIVE),
;    1228  0      ! All done executing the set of general commands
;    1229  0
;    1230  0  CRD$Level_4_Passed  : ADDRESSING_MODE (LONG_RELATIVE),
;    1231  0      . Print PASSED for the level 4 diagnostic that runs previous to ours
;    1232  0
;    1233  0  CRD$Load_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE),
;    1234  0      ! Load the AutoSizer
```

```
;   1235  0
;   1236  0  CRD$Load_General_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1237  0      ! Load a general command into the CLI data buffer
;   1238  0
;   1239  0  CRD$Load_Load_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1240  0      ! Load a LOAD command into the CLI buffer.
;   1241  0
;   1242  0  CRD$Load_Select_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1243  0      ! Load a SELECT command into the CLI buffer.
;   1244  0
;   1245  0  CRD$Load_Set_Event_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1246  0      ! Load a SET EVENT command into the CLI buffer.
;   1247  0
;   1248  0  CRD$Load_Set_Flags_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1249  0      ! Load a SET FLAGS command into the CLI buffer.
;   1250  0
;   1251  0  CRD$Load_Start_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1252  0      ! Load a START command into the CLI buffer.
;   1253  0
;   1254  0  CRD$Set_Up_Diagnostic  : ADDRESSING_MODE (LONG_RELATIVE),
;   1255  0      ! Perform setup for the next diagnostic (print testing started, etc).
;   1256  0
;   1257  0  CRD$Start_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE),
;   1258  0      ! Start the AutoSizer executing
;   1259  0
;   1260  0  !+
;   1261  0  ! From the CRDCTRLC module:
;   1262  0  !-
;   1263  0
;   1264  0  CRD$Set_Ctrl_C_Handlers  : JSB_R0_R1   NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1265  0
;   1266  0  CRD$Clear_Ctrl_C_Handlers : JSB_NO_REGS NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1267  0
;   1268  0  CRD$Disable_Ctrl_C  : JSB_NO_REGS NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1269  0
;   1270  0  CRD$Enable_Ctrl_C  : JSB_NO_REGS NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1271  0
;   1272  0  CRD$Set_Ctrl_C_Flag  : JSB_NO_REGS NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1273  0
;   1274  0  CRD$Clear_Ctrl_C_Flag  : JSB_NO_REGS NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1275  0
;   1276  0  CRD$Check_Ctrl_C  : JSB_NO_REGS         ADDRESSING_MODE (LONG_RELATIVE),
;   1277  0
;   1278  0  !+
;   1279  0  ! From the CRDDDB module:
;   1280  0  !-
;   1281  0
;   1282  0  CRD$Build_DDBs   : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1283  0      ! Build the queue of DDBs
;   1284  0
;   1285  0  CRD$Dispose   : ADDRESSING_MODE (LONG_RELATIVE),
;   1286  0      ! Dispose of non-paged pool memory.
;   1287  0
;   1288  0  CRD$Init_DDB_Status  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1289  0      ! Initialize the status bits in a DDB
```

```
;    1290  0
;    1291  0  CRD$New    : ADDRESSING_MODE (LONG_RELATIVE),
;    1292  0      ! Allocate a block of non-paged pool.
;    1293  0
;    1294  0  !+
;    1295  0  ! From the CRDGENCOM module:
;    1296  0  !-
;    1297  0
;    1298  0  CRD$Get_General_Command  : ADDRESSING_MODE (LONG_RELATIVE),
;    1299  0      ! Get the General_Commands command pointer
;    1300  0
;    1301  0  CRD$Init_General_Com_Table : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    1302  0      ! Initialize the General_Commands table
;    1303  0
;    1304  0  CRD$Print_Command_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    1305  0      ! Print the contents of the general command table
;    1306  0
;    1307  0  !+
;    1308  0  ! From the CRDHDWSUP module:
;    1309  0  !-
;    1310  0
;    1311  0  CRD$Build_Support_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    1312  0      !         [01]
;    1313  0      ! Initialize the Hardware Support table
;    1314  0
;    1315  0  CRD$Print_Hardware_Support : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    1316  0      ! Print the contents of the hardware support table
;    1317  0
;    1318  0  !+
;    1319  0  ! From the CRDHSACT module:
;    1320  0  !-
;    1321  0
;    1322  0  HS$Init_HS    : ADDRESSING_MODE (LONG_RELATIVE),
;    1323  0  HS$Advance_Diagnostic  : ADDRESSING_MODE (LONG_RELATIVE),
;    1324  0  HS$Load_General_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    1325  0  HS$Advance_General_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    1326  0  HS$Done_General_Coms  : ADDRESSING_MODE (LONG_RELATIVE),
;    1327  0  HS$Load_Load_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    1328  0  HS$Load_Set_Flags_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    1329  0  HS$Load_Set_Event_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    1330  0  HS$Load_Select_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    1331  0  HS$Load_Start_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    1332  0  HS$Change_Table   : ADDRESSING_MODE (LONG_RELATIVE),
;    1333  0  HS$Load_Error    : ADDRESSING_MODE (LONG_RELATIVE),
;    1334  0  HS$Start_Error    : ADDRESSING_MODE (LONG_RELATIVE),
;    1335  0  HS$Diagnostic_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    1336  0  HS$Preparation_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    1337  0  HS$Internal_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    1338  0
;    1339  0  CRD$Print_DS_Command  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    1340  0
;    1341  0  !+
;    1342  0  ! From the CRDINIT module:
;    1343  0  !-
;    1344  0
```

```
;    1345  0  CRD$Common_Initializations : ADDRESSING_MODE (LONG_RELATIVE),
;    1346  0      ! Initializations common to both CRD Menu Test and CRD Auto Test
;    1347  0
;    1348  0  CRD$CRD_Initialization  : ADDRESSING_MODE (LONG_RELATIVE),
;    1349  0      ! The initialization routine for the CRD Auto Test code.
;    1350  0
;    1351  0  !+
;    1352  0  ! From the CRDINTRFC module:
;    1353  0  !-
;    1354  0
;    1355  0  CRD$DS_Interface  : ADDRESSING_MODE (LONG_RELATIVE),
;    1356  0      ! The interface routine between the Resident-DS and CRD-Loadable
;    1357  0
;    1358  0  !+
;    1359  0  ! From the CRDMMACT module:
;    1360  0  !-
;    1361  0
;    1362  0  MM$Print_Menu_Heading  : ADDRESSING_MODE (LONG_RELATIVE),
;    1363  0  MM$Load_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE),
;    1364  0  MM$Set_Flags_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE),
;    1365  0  MM$Start_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE),
;    1366  0  MM$Build_DDBs  : ADDRESSING_MODE (LONG_RELATIVE),
;    1367  0  MM$Build_HSTs  : ADDRESSING_MODE (LONG_RELATIVE),
;    1368  0  MM$Done_Inits  : ADDRESSING_MODE (LONG_RELATIVE),
;    1369  0  MM$Print_Menu  : ADDRESSING_MODE (LONG_RELATIVE),
;    1370  0  MM$Change_Table  : ADDRESSING_MODE (LONG_RELATIVE),
;    1371  0  MM$Menu_Prompt  : ADDRESSING_MODE (LONG_RELATIVE),
;    1372  0  MM$Internal_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    1373  0  MM$AutoSizer_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    1374  0  MEN$Menu_Test_Internal_Error : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    1375  0  MEN$Undo_DCDT_Symbol  : ADDRESSING_MODE (LONG_RELATIVE),
;    1376  0
;    1377  0  !+
;    1378  0  ! From the CRDPTABLE module:
;    1379  0  !-
;    1380  0
;    1381  0  CRD$Find_PTable  : ADDRESSING_MODE (LONG_RELATIVE),
;    1382  0      ! Find a PTable containing a certain device name
;    1383  0
;    1384  0  CRD$Get_Device_Name  : ADDRESSING_MODE (LONG_RELATIVE),
;    1385  0      ! Return the ASCIC device name from a ptable
;    1386  0
;    1387  0  CRD$Print_PTables  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    1388  0      ! Print information in the PTables
;    1389  0
;    1390  0  !+
;    1391  0  ! From the CRDSTATE module:
;    1392  0  !-
;    1393  0
;    1394  0  CRD$Get_Action_Routine  : ADDRESSING_MODE (LONG_RELATIVE),
;    1395  0      ! Get the current action routine number
;    1396  0
;    1397  0  CRD$Get_State  : ADDRESSING_MODE (LONG_RELATIVE),
;    1398  0      ! Get the current state number
;    1399  0
```

```
;   1400  0  CRD$Init_State_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1401  0      ! Initialize the CRD$GW_State_Table
;   1402  0
;   1403  0  CRD$Turing_Machine  : ADDRESSING_MODE (LONG_RELATIVE),
;   1404  0      ! Step through the state table
;   1405  0
;   1406  0  CRD$Print_Action_Routine : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1407  0      ! Print a message saying what action routine number this is
;   1408  0
;   1409  0  !+
;   1410  0  ! From the CRDSTOP module:
;   1411  0  !-
;   1412  0
;   1413  0  CRD$Control_C_Handler  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1414  0      ! Handles Control-C's during the running of CRD
;   1415  0
;   1416  0  CRD$Stop   : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1417  0      ! Stop the CRD image.
;   1418  0
;   1419  0  CRD$Auto_Summary  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1420  0      ! Print an Auto Summary of Auto Test.
;   1421  0
;   1422  0  CRD$Screen_Pause  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1423  0      ! Stop video screen output until operator asks to continue. [02]
;   1424  0
;   1425  0  !+
;   1426  0  ! From the CRDTQACT module:
;   1427  0  !-
;   1428  0
;   1429  0  TQ$Init_TQ   : ADDRESSING_MODE (LONG_RELATIVE),
;   1430  0  TQ$Get_DDB   : ADDRESSING_MODE (LONG_RELATIVE),
;   1431  0  TQ$Change_Table   : ADDRESSING_MODE (LONG_RELATIVE),
;   1432  0  TQ$Done_Test_Queue  : ADDRESSING_MODE (LONG_RELATIVE),
;   1433  0  TQ$Internal_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;   1434  0
;   1435  0  !+
;   1436  0  ! From the CRDTYPCOD module:
;   1437  0  !-
;   1438  0
;   1439  0  CRD$Print_TypeCode_Name  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1440  0      ! Print a message for typecode number
;   1441  0
;   1442  0  !+
;   1443  0  ! From the MENTURING module:
;   1444  0  !-
;   1445  0
;   1446  0  MEN$Init_MM_State_Table  : NOVALUE ADDRESSING_MODE (LONG RELATIVE),
;   1447  0  MEN$Init_TQ_State_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1448  0  MEN$Init_HS_State_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;   1449  0
;   1450  0  MEN$Turing_Machine  : ADDRESSING_MODE (LONG_RELATIVE),
;   1451  0
;   1452  0  MEN$Print_Action_Routine : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    1453  0 %SBTTL 'External Longword Storage'
;    1454  0
;    1455  0 EXTERNAL
;    1456  0  !+
;    1457  0  ! From the CRDACT1 module:
;    1458  0  !-
;    1459  0
;    1460  0  CRD$GL_Total_Error_Count : ADDRESSING_MODE (LONG_RELATIVE),
;    1461  0      ! The total of all the Error_Count's below
;    1462  0
;    1463  0  CRD$GL_Other_Error_Count : ADDRESSING_MODE (LONG_RELATIVE),
;    1464  0      ! The total of other errors that occur (e.g., exceptions, command
;    1465  0      ! errors, etc.)
;    1466  0
;    1467  0  CRD$GL_Hard_Error_Count  : ADDRESSING_MODE (LONG_RELATIVE),
;    1468  0      ! A count of how many ERRHARD's were detected
;    1469  0
;    1470  0  CRD$GL_Soft_Error_Count  : ADDRESSING_MODE (LONG_RELATIVE),
;    1471  0      ! A count of how many ERRSOFT's were detected
;    1472  0
;    1473  0  CRD$GL_Sys_Error_Count  : ADDRESSING_MODE (LONG_RELATIVE),
;    1474  0      ! A count of how many ERRSYS's were detected
;    1475  0
;    1476  0  CRD$GL_Dev_Error_Count  : ADDRESSING_MODE (LONG_RELATIVE),
;    1477  0      ! A count of how many ERRDEV's were detected
;    1478  0
;    1479  0  CRD$GL_Prep_Error_Count  : ADDRESSING_MODE (LONG_RELATIVE),
;    1480  0      ! A count of how many ERRPREP's were detected
;    1481  0
;    1482  0  !+
;    1483  0  ! From the CRDDDB module:
;    1484  0  !-
;    1485  0
;    1486  0  CRD$GA_Current_Diagnostic : ADDRESSING_MODE (LONG_RELATIVE),
;    1487  0      ! A pointer to the DDB for which we are now running a diagnostic
;    1488  0
;    1489  0  CRD$GA_First_Diagnostic  : ADDRESSING_MODE (LONG_RELATIVE),
;    1490  0      ! A pointer to the first DDB in the queue (i.e., the head).
;    1491  0
;    1492  0  !+
;    1493  0  ! From the CRDGENCOM module:
;    1494  0  !-
;    1495  0
;    1496  0  CRD$GL_Current_General_Com : ADDRESSING_MODE (LONG_RELATIVE),
;    1497  0      ! The current general command pointer
;    1498  0
;    1499  0  !+
;    1500  0  ! From the CRDINIT module:
;    1501  0  !-
;    1502  0
;    1503  0  CRD$GL_Saved_DSA_Flags  : ADDRESSING_MODE (LONG_RELATIVE),
;    1504  0      ! To save the settings of the DSA flags
;    1505  0
;    1506  0  !+
;    1507  0  ! From the CRDINTRFC module:
```

```
;    1508  0  !-
;    1509  0
;    1510  0  CRD$GB_User_Prompt_Okay  : BYTE ADDRESSING_MODE (LONG_RELATIVE),
;    1511  0      ! This allows a User_Prompt typecode to be allowed to execute
;    1512  0
```

```
;    1513  0  !+
;    1514  0  ! From the CRDSTATE module:
;    1515  0  !-
;    1516  0
```

```
;    1517  0  CRD$GL_Current_Action  : ADDRESSING_MODE (LONG_RELATIVE),
;    1518  0      ! This is the current action routine number
;    1519  0
;    1520  0  CRD$GL_Current_State  : ADDRESSING_MODE (LONG_RELATIVE),
;    1521  0      ! This is the current state number
;    1522  0
;    1523  0  CRD$GL_Current_TypeCode  : ADDRESSING_MODE (LONG_RELATIVE),
;    1524  0      ! This is the current typecode number
;    1525  0
;    1526  0  !+
;    1527  0  ! From the CRDVECS module:
;    1528  0  !-
;    1529  0
;    1530  0  CRD$AL_CRD_Dispatch_Vectors : ADDRESSING_MODE (LONG_RELATIVE),
;    1531  0      ! Location of the beginning of the CRD dispatch vectors
;    1532  0
;    1533  0  CRD$Exit_DS   : ADDRESSING_MODE (LONG_RELATIVE),
;    1534  0      ! Location of the DSV$Exit routine
;    1535  0
;    1536  0  CRD$Print    : ADDRESSING_MODE (LONG_RELATIVE),
;    1537  0      ! Location of the DSX$Print routine
;    1538  0
;    1539  0  CRD$GA_DS_Flags    : ADDRESSING_MODE (LONG_RELATIVE),
;    1540  0      ! Location of the DS$GA_Flags longowrd
;    1541  0
;    1542  0  CRD$GA_DS_Ctrl_C_First  : ADDRESSING_MODE (LONG_RELATIVE),
;    1543  0      ! Location of the 1st ^C handler address
;    1544  0
;    1545  0  CRD$GA_DS_Ctrl_C_Second  : ADDRESSING_MODE (LONG_RELATIVE),
;    1546  0      ! Location of the 2nd ^C handler address
;    1547  0
;    1548  0  CRD$GA_PTable    : ADDRESSING_MODE (LONG_RELATIVE),
;    1549  0      ! Location of the DS$GA_PTable longword
;    1550  0
;    1551  0  CRD$Exe$AloNonPaged  : ADDRESSING_MODE (LONG_RELATIVE),
;    1552  0      ! Location of the EXE$AloNonPaged routine'
;    1553  0
;    1554  0  CRD$GA_User_Error_Text  : ADDRESSING_MODE (LONG_RELATIVE),
;    1555  0      ! Location of the MsgAdr parameter to the ERRxxxx routines
;    1556  0
;    1557  0  CRD$Scan$Numeric  : ADDRESSING_MODE (LONG_RELATIVE),
;    1558  0      ! Change ASCII numeric into longword numeric (from PARSE module)
;    1559  0
;    1560  0  CRD$Exe$DeaNonPaged  : ADDRESSING_MODE (LONG_RELATIVE),
;    1561  0      ! Deallocate non-paged pool memory.
;    1562  0
;    1563  0  CRD$GA_Begin    : ADDRESSING_MODE (LONG_RELATIVE),
;    1564  0      ! The address of the Begin label in the KERNEL module
;    1565  0
;    1566  0  CRD$GB_CRD_Debug  : BYTE ADDRESSING_MODE (LONG_RELATIVE),
;    1567  0      ! This is set from the VDS side, and passed as a dispatch vector.
;    1568  0
;    1569  0  CRD$DSR$Check_MenuTest_Off_Set : ADDRESSING_MODE (LONG_RELATIVE),
;    1570  0      ! This returns a 1 iff the correct bits are set in the R5 passed to the VDS
;    1571  0      ! to signal CRD Menu Test Offline
```

```
;    1572  0      !          [01]
;    1573  0  CRD$DSR$Check_AutoTest_Off_Set : ADDRESSING_MODE (LONG_RELATIVE),
;    1574  0      ! This returns a 1 iff the correct bits are set in the R5 passed to the VDS
;    1575  0      ! to signal CRD Auto Test Offline
;    1576  0      !          [01]
;    1577  0  !+
;    1578  0  ! From the MENSTATE module:
;    1579  0  !-
;    1580  0
;    1581  0  CRD$GL_Previous_State  : ADDRESSING_MODE (LONG_RELATIVE),
;    1582  0      ! The previous state number (in most cases)
;    1583  0
;    1584  0  CRD$GL_MM_Current_State  : ADDRESSING_MODE (LONG_RELATIVE),
;    1585  0      ! The current Main Menu state number
;    1586  0
;    1587  0  CRD$GL_TQ_Current_State  : ADDRESSING_MODE (LONG_RELATIVE),
;    1588  0      ! The current Test Queue state number
;    1589  0
;    1590  0  CRD$GL_HS_Current_State  : ADDRESSING_MODE (LONG_RELATIVE),
;    1591  0      ! The current Hardware Support state number
;    1592  0
;    1593  0  CRD$GL_TQ_Current_TQ_Entry : ADDRESSING_MODE (LONG_RELATIVE),
;    1594  0      ! The current entry being used in the Test Queue Table.
;    1595  0
;    1596  0  CRD$GL_HS_Current_HSB_Numb : ADDRESSING_MODE (LONG_RELATIVE),
;    1597  0      ! The number of the current Hardware Support Block in the concatenated
;    1598  0      ! Hardware Support Block Table.
;    1599  0
;    1600  0  CRD$GA_HS_Current_DDB_Ptr : ADDRESSING_MODE (LONG_RELATIVE),
;    1601  0      ! A pointer to the current DDB being tested
;    1602  0
;    1603  0  CRD$GA_HS_Current_HSB_Ptr : ADDRESSING_MODE (LONG_RELATIVE),
;    1604  0      ! A pointer to the current Hardware Support Block.
;    1605  0
;    1606  0  CRD$GB_Current_State_Table : BYTE ADDRESSING_MODE (LONG_RELATIVE);
;    1607  0      ! A pointer to the current Menu Test state table
```

```
;   1608  0 %SBTTL 'External Table Storage'
;   1609  0
;   1610  0 EXTERNAL
;   1611  0  !+
;   1612  0  ! From the CRDGENCOM module:
;   1613  0  !-
;   1614  0
;   1615  0  CRD$GA_General_Commands  : General_Command_Structure [CRD$K_Numb_General_Commands]
;   1616  0       ADDRESSING_MODE (LONG_RELATIVE),
;   1617  0       ! The CRD$GA_General_Commands table - contains those commands that
;   1618  0       ! are executed once before starting each diagnostic.
;   1619  0  !+
;   1620  0  ! From the CRDHDWSUP module:
;   1621  0  !-
;   1622  0
;   1623  0  CRD$GB_Base_System  : BYTE ADDRESSING_MODE (LONG_RELATIVE),
;   1624  0       !          [01]
;   1625  0  CRD$GA_Hdwr_Sprt_Table  : ADDRESSING_MODE (LONG_RELATIVE),
;   1626  0  CRD$AL_IDC_Hdwr_Sprt_Table :
;   1627  0   Hardware_Support_Structure [CRD$K_IDC_Numb_Diagnostics, CRD$K_Numb_Support_Entries]
;   1628  0   ADDRESSING_MODE (LONG_RELATIVE),!        [01]
;   1629  0
;   1630  0  CRD$AL_LESI_Hdwr_Sprt_Table :
;   1631  0   Hardware_Support_Structure [CRD$K_LESI_Numb_Diagnostics, CRD$K_Numb_Support_Entries]
;   1632  0   ADDRESSING_MODE (LONG_RELATIVE),!        [01]
;   1633  0
;   1634  0  CRD$AL_UDA_0_Hdwr_Sprt_Table :
;   1635  0   Hardware_Support_Structure [CRD$K_UDA_0_Numb_Diagnostics, CRD$K_Numb_Support_Entries]
;   1636  0   ADDRESSING_MODE (LONG_RELATIVE),!        [01]
;   1637  0
;   1638  0  CRD$AL_UDA_1_Hdwr_Sprt_Table :
;   1639  0   Hardware_Support_Structure [CRD$K_UDA_1_Numb_Diagnostics, CRD$K_Numb_Support_Entries]
;   1640  0   ADDRESSING_MODE (LONG_RELATIVE),!        [01]
;   1641  0
;   1642  0
;   1643  0  CRD$AL_UDA_2_Hdwr_Sprt_Table :
;   1644  0   Hardware_Support_Structure [CRD$K_UDA_0_Numb_Diagnostics, CRD$K_Numb_Support_Entries]
;   1645  0   ADDRESSING_MODE (LONG_RELATIVE),!        [01]
;   1646  0
;   1647  0  CRD$AL_UDA_3_Hdwr_Sprt_Table :
;   1648  0   Hardware_Support_Structure [CRD$K_UDA_1_Numb_Diagnostics, CRD$K_Numb_Support_Entries]
;   1649  0   ADDRESSING_MODE (LONG_RELATIVE),!        [01]
;   1650  0
;   1651  0  !+
;   1652  0  ! From the CRDMESSAG module:
;   1653  0  !-
;   1654  0
;   1655  0  CRD$GA_CRD_Message_Buffer :
;   1656  0   VECTOR [CRD$K_CRD_Message_Buffer_Len, BYTE]
;   1657  0   ADDRESSING_MODE (LONG_RELATIVE),
;   1658  0
;   1659  0  !+
;   1660  0  ! From the CRDSTATE module:
;   1661  0  !-
;   1662  0
```

```
;   1663  0  CRD$GW_State_Table :
;   1664  0   State_Table_Structure [CRD$K_Numb_TypeCodes, CRD$K_Numb_States]
;   1665  0   ADDRESSING_MODE (LONG_RELATIVE),
;   1666  0
;   1667  0  !+
;   1668  0  ! From the CRDStop Module:
;   1669  0
;   1670  0  CRD$GL_CRD_Debug_Vector :
;   1671  0   VECTOR [CRD$K_Total_Numb_Vectors, LONG]
;   1672  0   ADDRESSING_MODE (LONG_RELATIVE),
;   1673  0
;   1674  0  !+
;   1675  0  ! From the MENSTATE module:
;   1676  0  !-
;   1677  0
;   1678  0  CRD$GAW_MM_State_Table :
;   1679  0   MM_State_Table_Structure [CRD$K_Numb_TypeCodes, MM$K_Numb_States]
;   1680  0   ADDRESSING_MODE (LONG_RELATIVE),
;   1681  0
;   1682  0  CRD$GAW_TQ_State_Table :
;   1683  0   TQ_State_Table_Structure [CRD$K_Numb_TypeCodes, TQ$K_Numb_States]
;   1684  0   ADDRESSING_MODE (LONG_RELATIVE),
;   1685  0
;   1686  0  CRD$GAW_HS_State_Table :
;   1687  0   HS_State_Table_Structure [CRD$K_Numb_TypeCodes, HS$K_Numb_States]
;   1688  0   ADDRESSING_MODE (LONG_RELATIVE);
;   1689  0
```

```
;   1690  0 %SBTTL 'External ASCIC Bindings'
;   1691  0
;   1692  0 EXTERNAL
;   1693  0   !+
;   1694  0   ! From the CRDHDWSUP module:
;   1695  0   !-
;   1696  0
;   1697  0   CRD$GT_DS_LOAD_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1698  0   CRD$GT_DS_SET_FLAGS_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1699  0   CRD$GT_DS_SET_EVENT_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1700  0   CRD$GT_DS_SELECT_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1701  0   CRD$GT_DS_START_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;   1702  0
;   1703  0   !+
;   1704  0   ! From the CRDMESSAG module:
;   1705  0   !-
;   1706  0
;   1707  0   CRD$GT_New_Line        : ADDRESSING_MODE (LONG_RELATIVE),
;   1708  0   CRD$GT_Tab       : ADDRESSING_MODE (LONG_RELATIVE),
;   1709  0   CRD$GT_2_Tabs        : ADDRESSING_MODE (LONG_RELATIVE),
;   1710  0   CRD$GT_Space_AND_Space   : ADDRESSING_MODE (LONG_RELATIVE),
;   1711  0   CRD$GT_Slash       : ADDRESSING_MODE (LONG_RELATIVE),   ! [2]
;   1712  0   CRD$GT_Unknown       : ADDRESSING_MODE (LONG_RELATIVE),   ! [2]
;   1713  0 ! CRD$GT_Clear_Screen      : ADDRESSING_MODE (LONG_RELATIVE),   ! [2]
;   1714  0 ! CRD$GT_Clear_Screen_VT52 : ADDRESSING_MODE (LONG_RELATIVE),   ! [2]
;   1715  0
;   1716  0   CRD$GT_Menu        : ADDRESSING_MODE (LONG_RELATIVE),
;   1717  0   CRD$GT_Auto        : ADDRESSING_MODE (LONG_RELATIVE),
;   1718  0   CRD$GT_Which_CRD      : ADDRESSING_MODE (LONG_RELATIVE),
;   1719  0
;   1720  0   CRD$GT_Testing_Device      : ADDRESSING_MODE (LONG_RELATIVE),
;   1721  0   CRD$GT_Testing_Started      : ADDRESSING_MODE (LONG_RELATIVE),
;   1722  0   CRD$GT_RunTime       : ADDRESSING_MODE (LONG_RELATIVE),
;   1723  0   CRD$GT_Fail        : ADDRESSING_MODE (LONG_RELATIVE),
;   1724  0   CRD$GT_Pass        : ADDRESSING_MODE (LONG_RELATIVE),
;   1725  0
;   1726  0   CRD$GT_CRD_No_Errors     : ADDRESSING_MODE (LONG_RELATIVE),
;   1727  0   CRD$GT_CRD_Errors_Complete : ADDRESSING_MODE (LONG_RELATIVE),
;   1728  0   CRD$GT_CRD_Bad_Device     : ADDRESSING_MODE (LONG_RELATIVE),
;   1729  0   CRD$GT_CRD_Dev_UnTested  : ADDRESSING_MODE (LONG_RELATIVE),
;   1730  0   CRD$GT_CRD_Incomplete    : ADDRESSING_MODE (LONG_RELATIVE),
;   1731  0   CRD$GT_CRD_EVSBA_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;   1732  0   CRD$GT_CRD_EVSBB_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;   1733  0   CRD$GT_Internal_Error_Abort : ADDRESSING_MODE (LONG_RELATIVE),
;   1734  0   CRD$GT_Control_C_Abort      : ADDRESSING_MODE (LONG_RELATIVE),
;   1735  0
;   1736  0   CRD$GT_CRD_End_Message   : ADDRESSING_MODE (LONG_RELATIVE),
;   1737  0   CRD$GT_Exit_To_Console      : ADDRESSING_MODE (LONG_RELATIVE),
;   1738  0   CRD$GT_Exit_To_CLI       : ADDRESSING_MODE (LONG_RELATIVE),
;   1739  0   CRD$GT_Press_Return      : ADDRESSING_MODE (LONG_RELATIVE), !    [07]
;   1740  0   CRD$GT_Press_Return_MM      : ADDRESSING_MODE (LONG_RELATIVE), !    [07]
;   1741  0
;   1742  0   CRD$GT_Star_Line_Prefix_Message : ADDRESSING_MODE (LONG_RELATIVE),
;   1743  0   CRD$GT_Star_Line_Suffix_Message : ADDRESSING_MODE (LONG_RELATIVE),
;   1744  0   CRD$GT_Same_Line_Message  : ADDRESSING_MODE (LONG_RELATIVE),
```

```
;   1745  0  CRD$GT_New_Line_Message      : ADDRESSING_MODE (LONG_RELATIVE),
;   1746  0
;   1747  0  CRD$GT_Dev_Unavailable  : ADDRESSING_MODE (LONG_RELATIVE),
;   1748  0  CRD$GT_Dev_Unavailable_2 : ADDRESSING_MODE (LONG_RELATIVE),
;   1749  0  CRD$GT_Cannot_Load_Diag  : ADDRESSING_MODE (LONG_RELATIVE),
;   1750  0  CRD$GT_Start_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;   1751  0  CRD$GT_Prep_Error_Spaces : ADDRESSING_MODE (LONG_RELATIVE),
;   1752  0  CRD$GT_Internal_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;   1753  0  CRD$GT_AutoSizer_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;   1754  0  CRD$GT_Inv_Base_System  : ADDRESSING_MODE (LONG_RELATIVE), !    [01]
;   1755  0  CRD$GT_CRD_Fatal_Error  : ADDRESSING_MODE (LONG_RELATIVE), !    [01]
;   1756  0
;   1757  0  CRD$GT_DS_Command_Out  : ADDRESSING_MODE (LONG_RELATIVE),
;   1758  0
;   1759  0  CRD$GT_All_Failures  : ADDRESSING_MODE (LONG_RELATIVE), !    [02]
;   1760  0  CRD$GT_Init_All_Passes  : ADDRESSING_MODE (LONG_RELATIVE), !    [02]
;   1761  0  CRD$GT_All_Passes  : ADDRESSING_MODE (LONG_RELATIVE), !    [02]
;   1762  0  CRD$GT_All_Inc_Hdwr_Prep : ADDRESSING_MODE (LONG_RELATIVE), !    [02]
;   1763  0  CRD$GT_All_Others  : ADDRESSING_MODE (LONG_RELATIVE), !    [02]
;   1764  0  CRD$GT_Summary_Title  : ADDRESSING_MODE (LONG_RELATIVE), !    [02]
;   1765  0
;   1766  0  !+
;   1767  0  ! From the CRDMMACT module:
;   1768  0  !-
;   1769  0
;   1770  0  CRD$GT_AutoSizer_Name  : ADDRESSING_MODE (LONG_RELATIVE),
;   1771  0  CRD$GT_AutoSizer_Set_Flags : ADDRESSING_MODE (LONG_RELATIVE);
```

```
;   1772  0 %SBTTL 'LITERALS'
;   1773  0
; M 1774  0 MACRO $MEN_Literals =
; M 1775  0
; M 1776  0     LITERAL
; M 1777  0  $Equlst (MEN$K_TstCtg_,GLOBAL,0,1
; M 1778  0     ! CRD MENU device test category definitions
; M 1779  0    ,(NULL)  ! NULL Device test category
; M 1780  0    ,(CPU)  ! CPU device test category
; M 1781  0    ,(Memory) ! MEMORY device test category
; M 1782  0    ,(IO_Adaptor) ! I/O Adaptor device test category
; M 1783  0    ,(IO_Controller)! I/O Controller device test category
; M 1784  0    ,(IO_Unit) ! I/O Unit device test category
; M 1785  0    );
; M 1786  0
; M 1787  0     LITERAL
; M 1788  0  $Equlst (MEN$K_TstCla_,GLOBAL,0,1
; M 1789  0     ! CRD MENU device test class definitions
; M 1790  0    ,(NULL)  ! NULL Test Class
; M 1791  0    ,(CPU)  ! CPU test class
; M 1792  0    ,(Memory) ! Memory test class
; M 1793  0    ,(Disk)  ! Disk test class
; M 1794  0    ,(Tape)  ! Tape test class
; M 1795  0    ,(Comm)  ! Communication test class
; M 1796  0    ,(Real)  ! Realtime test class
; M 1797  0    ,(Term)  ! Terminal test class
; M 1798  0    ,(Printer) ! Printer test class
; M 1799  0    ,(Card)  ! Card Reader test class
; M 1800  0    ,(IO_Adaptor) ! I/O Adaptor test class
; M 1801  0    ,(ALL)  ! denotes all test classes
; M 1802  0    ,(Last_Entry)
; M 1803  0    );
; M 1804  0
; M 1805  0
; M 1806  0     LITERAL
; M 1807  0  $Equlst (MEN$K_DevTstPrep_,GLOBAL,0,1
; M 1808  0     ! CRD MENU message code for hardware test preparation.
; M 1809  0     ! Represents text that describes the
; M 1810  0     ! hardware test preparation required for
; M 1811  0     ! testing
; M 1812  0    ,(Null)  ! Null code
; M 1813  0    ,(1)  ! device test preparation code #1
; M 1814  0    ,(2)  ! device test preparation code #2
; M 1815  0    ,(3)  ! device test preparation code #3
; M 1816  0    ,(4)
; M 1817  0    ,(5)
; M 1818  0    ,(6)
; M 1819  0    ,(7)  !     ! [04]
; M 1820  0    );
; M 1821  0
; M 1822  0     LITERAL
; M 1823  0  $Equlst (MEN$K_FTMSel_,,1,1
; M 1824  0     ! CRD Functional Test
; M 1825  0     ! Main Menu selection LITERALs.
; M 1826  0    ,(Exit)  ! Exit selection
```

```
;  M 1827  0     ,(Prep)  ! Hardware Test Preparation typeout selection
;  M 1828  0     ,(Test)  ! Test Menu selection
;  M 1829  0     ,(System_Hdwr)  ! Identified System hardware typeout selection
;  M 1830  0     ,(FncTst_All) ! Functional Test all devices
;  M 1831  0     ,(Last_Entry)
;  M 1832  0     );
;  M 1833  0
;  M 1834  0       LITERAL
;  M 1835  0    $Equlst (MEN$K_CntlC_Sel_,,1,1
;  M 1836  0       ! Control-C Test Menu selection Literals
;  M 1837  0     ,(Exit)  ! Exit selection
;  M 1838  0     ,(Continue) ! Continue selection
;  M 1839  0     ,(FTMenu) ! Functional Test Menu selection
;  M 1840  0     ,(Last_Entry)
;  M 1841  0     );
;  M 1842  0
;  M 1843  0       LITERAL
;  M 1844  0    $Equlst (MEN$K_FTMRet_,,1,1
;  M 1845  0       ! CRD Functional Test Menu return codes
;  M 1846  0     ,(Exit)  ! Stop everything and exit
;  M 1847  0     ,(Menu)  ! Menu - do the Functional Test Menu
;  M 1848  0     ,(Test)  ! Test - Begin testing
;  M 1849  0     ,(Failure) ! Fatal Failure in Functional Test Menu
;  M 1850  0     );
;  M 1851  0
;  M 1852  0       LITERAL
;  M 1853  0    MEN$K_TMenu_TstAll_DevNumb = 1,  ! The device number for the
;  M 1854  0          ! Test Menu 'test all'
;  M 1855  0          ! choice
;  M 1856  0    MEN$K_SelDev_Numb_Start = 1,  ! Number to start allocating
;  M 1857  0          ! select device numbers
;  M 1858  0    MEN$K_TstTxt_UUTNam_Sz = 9,  ! Maximum size of hardware
;  M 1859  0          ! device name field in the
;  M 1860  0          ! test start output text
;  M 1861  0    MEN$K_TstTxt_TstCla_Sz = 9,  ! Maximum size of test class
;  M 1862  0          ! name field in the
;  M 1863  0          ! test start output text
;  M 1864  0    MEN$K_TstTxt_DevNam_Sz = 6,  ! Maximum size of generic
;  M 1865  0          ! device name field in the
;  M 1866  0          ! test start output text
;  M 1867  0
;  M 1868  0    MEN$K_Prep_Error_Text_Len = 38,  ! This is the length of the text part of the
;  M 1869  0          ! standard preparation error message.
;  M 1870  0
;  M 1871  0    MEN$GK_Test_Queue = 3;
;     1872  0       %;
;     1873  0
;     1874  0
;     1875  0  KEYWORDMACRO
;     1876  0
;     1877  0       $MEN_ASKSTR ( msg_adr,
;     1878  0      buf_adr,
;     1879  0      max_len=72,
;     1880  0      val_tab=0,
;     1881  0      def_adr=0,
```

```
; M 1882  0     dev_typ=0) =
; M 1883  0
; M 1884  0   (
; M 1885  0   BEGIN
; M 1886  0   LOCAL
; M 1887  0       Status;
; M 1888  0   $DS_BREAK;
; M 1889  0       ! Check for Control-C
; M 1890  0   Status = $DS_ASKSTR (msgadr=msg_adr,bufadr=buf_adr,maxlen=max_len,valtab=val_tab,defadr=def_adr,devtyp=dev_typ);
; M 1891  0       ! prompt for input
; M 1892  0   $DS_BREAK;
; M 1893  0       ! Check for Control-C
; M 1894  0
; M 1895  0   .Status
; M 1896  0   END
; M 1897  0   )
;    1898  0      %;
;    1899  0
```

```
;    1900  0 %SBTTL 'Externals for Menu Test'
;    1901  0 EXTERNAL
;    1902  0     !+
;    1903  0     ! From the CRD MENU Message Module (MENMSG)
;    1904  0     !-
;    1905  0
;    1906  0  MEN$GT_MenuTest_Begin    : ADDRESSING_MODE (LONG_RELATIVE),
;    1907  0
;    1908  0  MEN$GT_AutoSizer_Begin   : ADDRESSING_MODE (LONG_RELATIVE),
;    1909  0  MEN$GT_AutoSizer_End     : ADDRESSING_MODE (LONG_RELATIVE),
;    1910  0
;    1911  0  !+
;    1912  0  ! FUNCTIONAL TEST MAIN MENU
;    1913  0  !-
;    1914  0
;    1915  0  MEN$GT_FTMTitle    : ADDRESSING_MODE (LONG_RELATIVE),
;    1916  0  MEN$GT_FTMSel_Exit    : ADDRESSING_MODE (LONG_RELATIVE),
;    1917  0  MEN$GT_FTMSel_Prep    : ADDRESSING_MODE (LONG_RELATIVE),
;    1918  0  MEN$GT_FTMSel_Test    : ADDRESSING_MODE (LONG_RELATIVE),
;    1919  0  MEN$GT_FTMSel_System_Hdwr   : ADDRESSING_MODE (LONG_RELATIVE),
;    1920  0  MEN$GT_FTMSel_FncTst_All   : ADDRESSING_MODE (LONG_RELATIVE),
;    1921  0  MEN$GT_FTM_Choice    : ADDRESSING_MODE (LONG_RELATIVE),
;    1922  0  MEN$GT_FTMPrompt    : ADDRESSING_MODE (LONG_RELATIVE),
;    1923  0  MEN$GT_FTMenu_Selection   : ADDRESSING_MODE (LONG_RELATIVE),
;    1924  0
;    1925  0  !+
;    1926  0  ! TEST MENU
;    1927  0  !-
;    1928  0  MEN$GT_TMTitle    : ADDRESSING_MODE (LONG_RELATIVE),
;    1929  0  MEN$GT_TMenu_Selection    : ADDRESSING_MODE (LONG_RELATIVE),
;    1930  0  MEN$GT_TMenu_TstAll    : ADDRESSING_MODE (LONG_RELATIVE),
;    1931  0  MEN$GT_TM_Choice    : ADDRESSING_MODE (LONG_RELATIVE),
;    1932  0  MEN$GT_TMPrompt    : ADDRESSING_MODE (LONG_RELATIVE),
;    1933  0  MEN$GT_TM_Msg    : ADDRESSING_MODE (LONG_RELATIVE),     ! [02]
;    1934  0  MEN$GT_Press_Return_TM   : ADDRESSING_MODE (LONG_RELATIVE),     ! [02]
;    1935  0
;    1936  0  !+
;    1937  0  ! TEST CLASS NAMES
;    1938  0  !-
;    1939  0  MEN$GT_TstCla_CPU    : ADDRESSING_MODE (LONG_RELATIVE),
;    1940  0  MEN$GT_TstCla_Memory    : ADDRESSING_MODE (LONG_RELATIVE),
;    1941  0  MEN$GT_TstCla_Disk    : ADDRESSING_MODE (LONG_RELATIVE),
;    1942  0  MEN$GT_TstCla_Tape    : ADDRESSING_MODE (LONG_RELATIVE),
;    1943  0  MEN$GT_TstCla_Comm    : ADDRESSING_MODE (LONG_RELATIVE),
;    1944  0  MEN$GT_TstCla_Real    : ADDRESSING_MODE (LONG_RELATIVE),
;    1945  0  MEN$GT_TstCla_Term    : ADDRESSING_MODE (LONG_RELATIVE),
;    1946  0  MEN$GT_TstCla_Printer    : ADDRESSING_MODE (LONG_RELATIVE),
;    1947  0  MEN$GT_TstCla_Card    : ADDRESSING_MODE (LONG_RELATIVE),
;    1948  0  MEN$GT_TstCla_IOAdap    : ADDRESSING_MODE (LONG_RELATIVE),
;    1949  0  MEN$GT_TstCla_All    : ADDRESSING_MODE (LONG_RELATIVE),
;    1950  0  MEN$GT_TstCla_NullName    : ADDRESSING_MODE (LONG_RELATIVE),
;    1951  0
;    1952  0  !+
;    1953  0  ! Hardware Device Test Preparation messages
;    1954  0  !-
```

```
;    1955  0
;    1956  0  MEN$GT_DevTstPrep_Null   :ADDRESSING_MODE (LONG_RELATIVE),
;    1957  0  MEN$GT_DevTstPrep_1    :ADDRESSING_MODE (LONG_RELATIVE),
;    1958  0  MEN$GT_DevTstPrep_2    :ADDRESSING_MODE (LONG_RELATIVE),
;    1959  0  MEN$GT_DevTstPrep_3    :ADDRESSING_MODE (LONG_RELATIVE),
;    1960  0  MEN$GT_DevTstPrep_4    :ADDRESSING_MODE (LONG_RELATIVE),
;    1961  0  MEN$GT_DevTstPrep_5    :ADDRESSING_MODE (LONG_RELATIVE),
;    1962  0  MEN$GT_DevTstPrep_6    :ADDRESSING_MODE (LONG_RELATIVE),    ! [02]
;    1963  0  MEN$GT_DevTstPrep_7    :ADDRESSING_MODE (LONG_RELATIVE),    ! [05]
;    1964  0  MEN$GT_DevTstPrep_Name   :ADDRESSING_MODE (LONG_RELATIVE),
;    1965  0  MEN$GT_DevTstPrep_Text   :ADDRESSING_MODE (LONG_RELATIVE),
;    1966  0
;    1967  0  !+
;    1968  0  ! Print system Hardware text
;    1969  0  !-
;    1970  0
;    1971  0  MEN$GT_SysHdwr_Title  : ADDRESSING_MODE (LONG_RELATIVE),
;    1972  0  MEN$GT_System_Hdwr_Text  : ADDRESSING_MODE (LONG_RELATIVE),
;    1973  0  MEN$GT_Supported    : ADDRESSING_MODE (LONG_RELATIVE),
;    1974  0  MEN$GT_NoSupport    : ADDRESSING_MODE (LONG_RELATIVE),
;    1975  0  MEN$GT_End_of_List  : ADDRESSING_MODE (LONG_RELATIVE),    ! [02]
;    1976  0  !+
;    1977  0  ! Warning messages
;    1978  0  !-
;    1979  0
;    1980  0  MEN$GT_InvOperInput    : ADDRESSING_MODE (LONG_RELATIVE),
;    1981  0  MEN$GT_Call_FldSrv    : ADDRESSING_MODE (LONG_RELATIVE),
;    1982  0  MEN$GT_Failed_Hdwr_Name  : ADDRESSING_MODE (LONG_RELATIVE),
;    1983  0  MEN$GT_Prep_Error_Text   : ADDRESSING_MODE (LONG_RELATIVE),
;    1984  0  MEN$GT_Prep_Error_Text_No_User : ADDRESSING_MODE (LONG_RELATIVE),
;    1985  0  MEN$GT_ScrMedia_Msg    : ADDRESSING_MODE (LONG_RELATIVE),
;    1986  0  MEN$GT_ScrMedia_Hdwr    : ADDRESSING_MODE (LONG_RELATIVE),
;    1987  0  MEN$GT_ScrMedia_Hdwr_End : ADDRESSING_MODE (LONG_RELATIVE),
;    1988  0  MEN$GT_Failed_ONL_Autosizer : ADDRESSING_MODE (LONG_RELATIVE),
;    1989  0
;    1990  0  !+
;    1991  0  ! Info Messages
;    1992  0  !-
;    1993  0
;    1994  0  MEN$GT_FuncTest_Begin    : ADDRESSING_MODE (LONG_RELATIVE),
;    1995  0  MEN$GT_CtrlC_FncTst    : ADDRESSING_MODE (LONG_RELATIVE),
;    1996  0  MEN$GT_FncTst_Cmpl_NoErr  : ADDRESSING_MODE (LONG_RELATIVE),
;    1997  0  MEN$GT_FncTst_Cmpl_Err    : ADDRESSING_MODE (LONG_RELATIVE),
;    1998  0  MEN$GT_FncTst_InCmpl_NoErr  : ADDRESSING_MODE (LONG_RELATIVE),
;    1999  0  MEN$GT_FncTst_InCmpl_Err  : ADDRESSING_MODE (LONG_RELATIVE),
;    2000  0  MEN$GT_Exit_To_VDS    : ADDRESSING_MODE (LONG_RELATIVE),
;    2001  0  MEN$GT_MenuTest_Aborted  : ADDRESSING_MODE (LONG_RELATIVE),
;    2002  0  MEN$GT_Operator_Abort  : ADDRESSING_MODE (LONG_RELATIVE),
;    2003  0  MEN$GT_Operator_Stop  : ADDRESSING_MODE (LONG_RELATIVE),
;    2004  0  MEN$GT_Assistance_Msg    : ADDRESSING_MODE (LONG_RELATIVE),
;    2005  0  MEN$GT_MenuTest_End_Message  : ADDRESSING_MODE (LONG_RELATIVE),
;    2006  0  MEN$GT_FncTst_TestStrt_Text  : ADDRESSING_MODE (LONG_RELATIVE),
;    2007  0  MEN$GT_Runtime_Total    : ADDRESSING_MODE (LONG_RELATIVE),
;    2008  0  MEN$GT_NULL     : ADDRESSING_MODE (LONG_RELATIVE),
;    2009  0  MEN$GT_ScrMedia_Prompt  : ADDRESSING_MODE (LONG_RELATIVE),
```

```
;    2010  0
;    2011  0  !+
;    2012  0  ! Error Messages
;    2013  0  !-
;    2014  0
;    2015  0  MEN$GT_NoAll_NonPgMem    : ADDRESSING_MODE (LONG_RELATIVE),
;    2016  0  MEN$GT_Support_File_Not_Found : ADDRESSING_MODE (LONG_RELATIVE),
;    2017  0  MEN$GT_Support_File_Load_Err : ADDRESSING_MODE (LONG_RELATIVE),
;    2018  0
;    2019  0  !+
;    2020  0  ! Control-C Menu Text
;    2021  0  !-
;    2022  0
;    2023  0  MEN$GT_CntlC_Menu_Title  : ADDRESSING_MODE (LONG_RELATIVE),
;    2024  0  MEN$GT_CntlC_Menu_Selection  : ADDRESSING_MODE (LONG_RELATIVE),
;    2025  0  MEN$GT_CntlC_Sel_Exit    : ADDRESSING_MODE (LONG_RELATIVE),
;    2026  0  MEN$GT_CntlC_Sel_Continue  : ADDRESSING_MODE (LONG_RELATIVE),
;    2027  0  MEN$GT_CntlC_Sel_FTMenu  : ADDRESSING_MODE (LONG_RELATIVE),
;    2028  0  MEN$GT_CntlCM_Choice    : ADDRESSING_MODE (LONG_RELATIVE),
;    2029  0  MEN$GT_CntlC_Menu_Prompt  : ADDRESSING_MODE (LONG_RELATIVE),
;    2030  0  MEN$GT_CntlC_Continuing  : ADDRESSING_MODE (LONG_RELATIVE),
;    2031  0
;    2032  0  MEN$GT_MenFuncSup_File  : VECTOR [, BYTE] ADDRESSING_MODE (LONG_RELATIVE),
;    2033  0  MEN$GT_File_Default  : VECTOR [, BYTE] ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    2034  0 %SBTTL 'External Storage for Menu Test'
;    2035  0
;    2036  0 EXTERNAL
;    2037  0  MEN$GB_Test_In_Progress  : BYTE ADDRESSING_MODE (LONG_RELATIVE),
;    2038  0  MEN$GB_FncTst_Init  : BYTE ADDRESSING_MODE (LONG_RELATIVE);
;    2039  0 %SBTTL 'External Menu Test Tables'
;    2040  0
;    2041  0 LITERAL
;    2042  0  MEN$K_OperInput_Buffer_Size = 256;
;    2043  0
;    2044  0 MACRO
; M  2045  0  $MEN_OperInput_Buffer_ATTR =
; M  2046  0   VECTOR [MEN$K_OperInput_Buffer_Size, BYTE]
;    2047  0  %;
;    2048  0
;    2049  0 EXTERNAL
;    2050  0  MEN$GT_OperInput_Buffer : $MEN_OperInput_Buffer_ATTR
;    2051  0       ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    2052  0 %SBTTL 'External Routines for Menu Test'
;    2053  0
;    2054  0 EXTERNAL ROUTINE
;    2055  0  !+
;    2056  0  ! from the MENTSTINI Module
;    2057  0  !-
;    2058  0
;    2059  0  MEN$Menu_FncTst_Init  : ADDRESSING_MODE (LONG_RELATIVE),
;    2060  0  MEN$Build_DDB  : ADDRESSING_MODE (LONG_RELATIVE),
;    2061  0  MEN$Determine_Support  : ADDRESSING_MODE (LONG_RELATIVE),
;    2062  0  MEN$Specify_SelectNo  : ADDRESSING_MODE (LONG_RELATIVE),
;    2063  0  MEN$Allocate_Memory  : ADDRESSING_MODE (LONG_RELATIVE),
;    2064  0  MEN$Load_File  : ADDRESSING_MODE (LONG_RELATIVE),
;    2065  0  MEN$Find_SupBlk  : ADDRESSING_MODE (LONG_RELATIVE),
;    2066  0  MEN$Dump_TstQ_HST  : ADDRESSING_MODE (LONG_RELATIVE),
;    2067  0  MEN$Menu_Cleanup  : ADDRESSING_MODE (LONG_RELATIVE),
;    2068  0
;    2069  0  !+
;    2070  0  ! from the MENPROMPT Module
;    2071  0  !-
;    2072  0
;    2073  0  MEN$FuncTest_Menu  : ADDRESSING_MODE (LONG_RELATIVE),
;    2074  0  MEN$Test_Menu  : ADDRESSING_MODE (LONG_RELATIVE),
;    2075  0  MEN$Get_TstCla_Name  : ADDRESSING_MODE (LONG_RELATIVE),
;    2076  0  MEN$FndDDB_Tcl_SDN  : ADDRESSING_MODE (LONG_RELATIVE),
;    2077  0  MEN$FndDDB_Tcl  : ADDRESSING_MODE (LONG_RELATIVE),
;    2078  0  MEN$Print_DevTstPrep  : ADDRESSING_MODE (LONG_RELATIVE),
;    2079  0  MEN$Print_System_Hdwr  : ADDRESSING_MODE (LONG_RELATIVE),
;    2080  0  MEN$FTMTbl_Init  : ADDRESSING_MODE (LONG_RELATIVE),
;    2081  0  MEN$PrepTbl_Init  : ADDRESSING_MODE (LONG_RELATIVE),
;    2082  0
;    2083  0  !+
;    2084  0  ! from the MENTSTQ Module
;    2085  0  !-
;    2086  0
;    2087  0  MEN$BldTstQ_FncTst_All  : ADDRESSING_MODE (LONG_RELATIVE),
;    2088  0  MEN$BldTstQ_FncTst_Dev  : ADDRESSING_MODE (LONG_RELATIVE),
;    2089  0
;    2090  0  !+
;    2091  0  ! from the MENTST Module
;    2092  0  !-
;    2093  0
;    2094  0  MEN$FncTst_CmplErr_Status  : ADDRESSING_MODE (LONG_RELATIVE),
;    2095  0  MEN$FncTst_TestStrt_Text  : ADDRESSING_MODE (LONG_RELATIVE),
;    2096  0  MEN$Print_Failed_Hdwr  : ADDRESSING_MODE (LONG_RELATIVE),
;    2097  0  MEN$Print_Runtime_Total  : ADDRESSING_MODE (LONG_RELATIVE),
;    2098  0  MEN$CVT_Time_ASC_D  : ADDRESSING_MODE (LONG_RELATIVE),
;    2099  0  MEN$Scan_Numeric_ASC_D  : ADDRESSING_MODE (LONG_RELATIVE),
;    2100  0  MEN$Preparation_Err_Text : ADDRESSING_MODE (LONG_RELATIVE),
;    2101  0  MEN$Scratch_Media_Msg  : ADDRESSING_MODE (LONG_RELATIVE),
;    2102  0  MEN$Menu_Summary  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    2103  0
;    2104  0  !+
;    2105  0  ! from the MENCNTLC Module
;    2106  0  !
```

```
;   2107  0
;   2108  0  MEN$CntIC_Tbl_Init  : ADDRESSING_MODE (LONG_RELATIVE),
;   2109  0  MEN$CntICInit_Tbl_Init   : ADDRESSING_MODE (LONG_RELATIVE),
;   2110  0  MEN$CntIC_Menu    : ADDRESSING_MODE (LONG_RELATIVE);
```

```
;   2111  0 %SBTTL 'Doubly-linked list queue header field definitions'
;   2112  0
;   2113  0 LITERAL
;   2114  0     MEN$K_Queue_Link_Size = 8;
;   2115  0
;   2116  0 FIELD
;   2117  0     $MEN_Queue_Link_FLD =
;   2118  0  SET
;   2119  0  Queue$L_Flink = [0,0,32,0],
;   2120  0  Queue$L_Blink = [4,0,32,0]
;   2121  0  TES;
;   2122  0
;   2123  0 MACRO
; M 2124  0     $MEN_Queue_Link_ATTR =
; M 2125  0  BLOCK [MEN$K_Queue_Link_Size,BYTE]
; M 2126  0  FIELD ($MEN_Queue_Link_FLD)
;   2127  0     %;
;   2128  0
;   2129  0 EXTERNAL
;   2130  0  MEN$GA_DDB_Head : $MEN_Queue_Link_ATTR
;   2131  0      ADDRESSING_MODE (LONG_RELATIVE);
;   2132  0
;   2133  0
```

```
;   2134  0 %SBTTL 'Support Database Identification Block field definitions'
;   2135  0 LITERAL
;   2136  0     MEN$K_DBaseID_Size = 24;
;   2137  0
;   2138  0 FIELD
;   2139  0     $MEN_DBaseID_FLD =
;   2140  0  SET
;   2141  0  DBaseID$L_Size  = [0,0,32,0], ! Size in longwords of support
;   2142  0      ! Data Base ID Block
;   2143  0  DBaseID$T_Version = [4,0,0,0]
;   2144  0      ! Version # ASCIC in the form
;   2145  0      ! 'Version = #'
;   2146  0  TES;
;   2147  0
;   2148  0 MACRO
; M 2149  0     $MEN_DBaseID_ATTR =
; M 2150  0  BLOCK [MEN$K_DBaseID_Size,BYTE]
; M 2151  0  FIELD ($MEN_DBaseID_FLD)
;   2152  0     %;
;   2153  0
```

```
;   2154  0 %SBTTL 'Support Block field definitions'
;   2155  0 LITERAL
;   2156  0     MEN$K_SupBlk_Size = 23;
;   2157  0
;   2158  0 FIELD
;   2159  0     $MEN_SupBlk_FLD =
;   2160  0  SET
;   2161  0  SupBlk$L_SupBlk_Size  = [0,0,32,0], ! Size in longwords of support block
;   2162  0  SupBlk$T_Device_Name  = [4,0,0,0], ! Hardware Device Name ASCIC
;   2163  0  SupBlk$B_Test_Category  = [16,0,8,0], ! Device Test Category Code
;   2164  0        ! defined by MEN$K_TstCtg_xxx
;   2165  0  SupBlk$B_Test_Class  = [17,0,8,0], ! Device Test Class code ,
;   2166  0        ! defined by MEN$K_TstCla_x
;   2167  0  SupBlk$B_TstCnfg_No  = [18,0,8,0], ! Number of Test Configuraion
;   2168  0        ! blocks in this Support Block
;   2169  0  SupBlk$W_Status  = [19,0,16,0], ! Support Block Status Word.
;   2170  0  SupBlk$V_Status_Invalid = [19,0,1,0], ! Support Block Invalid bit.
;   2171  0        ! Indicates that the support
;   2172  0        ! block is invalid.
;   2173  0        ! 0 or False = Valid.
;   2174  0        ! 1 or True = Invalid
;   2175  0  SupBlk$W_Level    = [21,0,16,0], ! Support Block Diag Level
;   2176  0  SupBlk$B_StrtFrst_CnfgBlk = [23,0,0,0] ! Start of first Test Configuration block
;   2177  0  TES;
;   2178  0
;   2179  0 MACRO
; M 2180  0     $MEN_SupBlk_ATTR =
; M 2181  0  BLOCK [MEN$K_SupBlk_Size,BYTE]
; M 2182  0  FIELD ($MEN_SupBlk_FLD)
;   2183  0     %;
;   2184  0
```

```
;    2185  0 %SBTTL 'Test Configuration Block field definitions'
;    2186  0
;    2187  0
;    2188  0 LITERAL
;    2189  0     MEN$K_TstCnfg_Size = 4;
;    2190  0
;    2191  0 FIELD
;    2192  0     $MEN_TstCnfg_FLD =
;    2193  0  SET
;    2194  0  TstCnfg$B_Prep = [0,0,8,0],  ! A device test preparation code
;    2195  0        ! representing text describing
;    2196  0        ! the hardware device test
;    2197  0        ! preparation required
;    2198  0  TstCnfg$B_Diagblk_No = [1,0,8,0], ! Number of concatenated
;    2199  0        ! Diagnostic Blocks
;    2200  0        ! in this Test Configuration block
;    2201  0  TstCnfg$W_Status = [2,0,16,0],  ! Test Configuration Status word
;    2202  0  TstCnfg$V_Status_Scratch =
;    2203  0        [2,0,1,0],  ! Scratch media required bit.
;    2204  0        ! Indicates that media required
;    2205  0        ! for testing hardware must
;    2206  0        ! be scratch ( must not
;    2207  0        ! contain useable data).
;    2208  0        ! 0 or False = Non-Destructive
;    2209  0        ! 1 or True = Destructive
;    2210  0  TstCnfg$B_StrtFrst_DiagBlk =
;    2211  0     [4,0,0,0] ! Start of first Diagnostic Block
;    2212  0  TES;
;    2213  0
;    2214  0 MACRO
; M  2215  0     $MEN_TstCnfg_ATTR =
; M  2216  0  BLOCK [MEN$K_TstCnfg_Size,BYTE]
; M  2217  0  FIELD ($MEN_TstCnfg_FLD)
;    2218  0     %;
```

```
;   2219  0 %SBTTL 'Diagnostic Block field definitions'
;   2220  0 LITERAL
;   2221  0     MEN$K_DiagBlk_Size = 2;
;   2222  0
;   2223  0 FIELD
;   2224  0     $MEN_DiagBlk_FLD =
;   2225  0  SET
;   2226  0  DiagBlk$W_Record_No = [0,0,16,0], ! Number of records in Diagnostic
;   2227  0        ! Block
;   2228  0  DiagBlk$B_Frst_Record = [2,0,0,0]  ! First record
;   2229  0  TES;
;   2230  0
;   2231  0 MACRO
; M 2232  0     $MEN_DiagBlk_ATTR =
; M 2233  0  BLOCK [MEN$K_DiagBlk_Size,BYTE]
; M 2234  0  FIELD ($MEN_DiagBlk_FLD)
;   2235  0     %;
```

```
;    2236  0 %SBTTL 'ASCII Support Database'
;    2237  0
;    2238  0 LITERAL
;    2239  0  MEN$GK_FuncSup_DatBase = 16; ! 16 Bytes long
;    2240  0
;    2241  0 FIELD
;    2242  0  $MEN_FuncSup_DatBase_Fld =
;    2243  0   SET
;    2244  0   MEN$V_FuncSup_DatBase_Size = [0, 0, 32, 0],
;    2245  0      ! Size in bytes of database
;    2246  0   MEN$V_FuncSup_DatBase_Ptr = [4, 0, 32, 0],
;    2247  0      ! Address pointer to first
;    2248  0      ! device support block
;    2249  0   MEN$V_FuncSup_DatBase_Total_Siz = [8, 0, 32, 0],
;    2250  0      ! Size in bytes of allocated
;    2251  0      ! memory
;    2252  0   MEN$V_FuncSup_DatBase_Start = [12, 0, 32, 0]
;    2253  0      ! Address pointer to start of
;    2254  0      ! database.
;    2255  0   TES;
;    2256  0
;    2257  0 MACRO
; M  2258  0  $MEN_FuncSup_DatBase_ATTR =
; M  2259  0   BLOCK [MEN$GK_FuncSup_DatBase, BYTE]
; M  2260  0   FIELD ($MEN_FuncSup_DatBase_Fld)
;    2261  0  %;
;    2262  0
;    2263  0 EXTERNAL
;    2264  0  MEN$GA_FuncSup_DatBase : $MEN_FuncSup_DatBase_ATTR
;    2265  0      ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    2266  0 %SBTTL 'Functional Test Main Menu Definitions'
;    2267  0
;    2268  0 !+
;    2269  0 ! FUNCTIONAL TEST MAIN MENU Table entry field definitions
;    2270  0 !-
;    2271  0
;    2272  0 LITERAL
;    2273  0     MEN$GK_FTMTbl_Entries   = 5,
;    2274  0     MEN$K_FTMTBL_Entry_Size = 6;
;    2275  0
;    2276  0 FIELD
;    2277  0     $MEN_FTMTbl_Entry_FLD =
;    2278  0  SET
;    2279  0  FTMTbl$B_Entry_Alpha = [0,0,8,0], ! Menu selection choice
;    2280  0  FTMTbl$B_Entry_FTMSel = [1,0,8,0], ! Functional Test Main
;    2281  0        ! Menu selection code
;    2282  0  FTMTbl$L_Entry_Text   = [2,0,32,0] ! pointer to Menu selection text ASCIC
;    2283  0  TES;
;    2284  0
;    2285  0 MACRO
; M  2286  0     $MEN_FTMTbl_Entry_ATTR =
; M  2287  0  BLOCK [MEN$K_FTMTbl_Entry_Size,BYTE]
; M  2288  0  FIELD ($MEN_FTMTbl_Entry_FLD)
;    2289  0     %;
;    2290  0
;    2291  0 MACRO
; M  2292  0     $MEN_FTMTbl_ATTR =
; M  2293  0  BLOCKVECTOR [MEN$GK_FTMTbl_Entries, MEN$K_FTMTbl_Entry_Size, BYTE]
; M  2294  0  FIELD ($MEN_FTMTbl_Entry_FLD)
;    2295  0     %;
;    2296  0
;    2297  0 EXTERNAL
;    2298  0  MEN$GA_FTMTbl : $MEN_FTMTbl_ATTR
;    2299  0    ADDRESSING_MODE (LONG_RELATIVE);
;    2300  0
;    2301  0 !+
;    2302  0 ! TEST MENU Table entry field definitions
;    2303  0 !-
;    2304  0
;    2305  0 LITERAL
;    2306  0     MEN$GK_TMTbl_Entries   = 11,
;    2307  0     MEN$K_TMTbl_Entry_Size = 2;
;    2308  0
;    2309  0 FIELD
;    2310  0     $MEN_TMTbl_Entry_FLD =
;    2311  0  SET
;    2312  0  TMTbl$B_Entry_Alpha = [0,0,8,0], ! Menu selection choice
;    2313  0  TMTbl$B_Entry_TstCla = [1,0,8,0] ! Test Class code
;    2314  0  TES;
;    2315  0
;    2316  0 MACRO
; M  2317  0     $MEN_TMTbl_Entry_ATTR =
; M  2318  0  BLOCK [MEN$K_TMTbl_Entry_Size,BYTE]
; M  2319  0  FIELD ($MEN_TMTbl_Entry_FLD)
;    2320  0     %;
```

```
;    2321  0
;    2322  0 MACRO
; M  2323  0      $MEN_TMTbl_ATTR =
; M  2324  0   BLOCKVECTOR [MEN$GK_TMTbl_Entries, MEN$K_TMTbl_Entry_Size, BYTE]
; M  2325  0   FIELD ($MEN_TMTbl_Entry_FLD)
;    2326  0      %;
;    2327  0
;    2328  0 EXTERNAL
;    2329  0   MEN$GA_TMTbl : $MEN_TMTbl_ATTR
;    2330  0            ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    2331  0 %SBTTL 'Device Preparation Table Definitions'
;    2332  0
;    2333  0 !+
;    2334  0 ! DEVICE TEST PREPARATION Table entry field definitions
;    2335  0 !-
;    2336  0
;    2337  0 LITERAL
;    2338  0     MEN$GK_PrepTbl_Entries   = 8,
;    2339  0     MEN$K_PrepTbl_Entry_Size = 5;
;    2340  0
;    2341  0 FIELD
;    2342  0     $MEN_PrepTbl_Entry_FLD =
;    2343  0  SET
;    2344  0  PrepTbl$B_Entry_Code = [0,0,8,0], ! Preparation code
;    2345  0  PrepTbl$L_Entry_Text = [1,0,32,0] ! Preparation text ASCIC pointer
;    2346  0  TES;
;    2347  0
;    2348  0 MACRO
; M  2349  0     $MEN_PrepTbl_Entry_ATTR =
; M  2350  0  BLOCK [MEN$K_PrepTbl_Entry_Size,BYTE]
; M  2351  0  FIELD ($MEN_PrepTbl_Entry_FLD)
;    2352  0     %;
;    2353  0
;    2354  0 MACRO
; M  2355  0     $MEN_PrepTbl_ATTR =
; M  2356  0  BLOCKVECTOR [MEN$GK_PrepTbl_Entries, MEN$K_PrepTbl_Entry_Size, BYTE]
; M  2357  0  FIELD ($MEN_PrepTbl_Entry_FLD)
;    2358  0     %;
;    2359  0
;    2360  0 EXTERNAL
;    2361  0  MEN$GA_PrepTbl : $MEN_PrepTbl_ATTR
;    2362  0     ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    2363  0 %SBTTL 'Control-C Menu Table Definitions'
;    2364  0
;    2365  0 !
;    2366  0 !+
;    2367  0 ! CONTROL-C MENU Table entry field definitions
;    2368  0 !-
;    2369  0
;    2370  0 LITERAL
;    2371  0     MEN$K_CntlCTbl_Entry_Size = 6;
;    2372  0
;    2373  0 FIELD
;    2374  0     $MEN_CntlCTbl_Entry_FLD =
;    2375  0  SET
;    2376  0  CntlCTbl$B_Entry_Alpha = [0,0,8,0], ! Menu selection choice
;    2377  0  CntlCTbl$B_Entry_CntlCSel = [1,0,8,0], ! Control-C Test Menu selection code
;    2378  0  CntlCTbl$L_Entry_Text   = [2,0,32,0] ! pointer to Menu selection text ASCIC
;    2379  0  TES;
;    2380  0
;    2381  0 MACRO
; M  2382  0     $MEN_CntlCTbl_Entry_ATTR =
; M  2383  0  BLOCK [MEN$K_CntlCTbl_Entry_Size,BYTE]
; M  2384  0  FIELD ($MEN_CntlCTbl_Entry_FLD)
;    2385  0     %;
```

```
;   2386  0 %SBTTL 'Definitions for accessing the Test Queue Table'
;   2387  0
;   2388  0 LITERAL
;   2389  0  MEN$K_Test_Queue_Ptr_Blk_Size = 12;
;   2390  0       ! 12 Bytes long
;   2391  0
;   2392  0 FIELD
;   2393  0  MEN$V_Test_Queue_Ptr_Blk_Flds =
;   2394  0   SET
;   2395  0   MEN$V_TQ_Table_Numb_Entries = [0, 0, 32, 0],
;   2396  0   MEN$V_TQ_Table_Ptr     = [4, 0, 32, 0],
;   2397  0   MEN$V_TQ_Table_Size     = [8, 0, 32, 0]
;   2398  0   TES;
;   2399  0
;   2400  0 MACRO
; M 2401  0  $MEN_Test_Queue_ATTR =
; M 2402  0   BLOCK [MEN$K_Test_Queue_Ptr_Blk_Size, BYTE]
; M 2403  0   FIELD  (MEN$V_Test_Queue_Ptr_Blk_Flds)
;   2404  0  %;
;   2405  0
;   2406  0 EXTERNAL
;   2407  0  MEN$GA_Test_Queue : $MEN_Test_Queue_ATTR
;   2408  0        ADDRESSING_MODE (LONG_RELATIVE);
```

```
;   2409  0 %SBTTL 'Definitions for Console Terminal Characteristics'
;   2410  0
;   2411  0 LITERAL
;   2412  0  CRD$K_Termchar_Size = 8; ! 8 bytes long
;   2413  0
;   2414  0 FIELD
;   2415  0  $CRD_Termchar_FLD =
;   2416  0      SET
;   2417  0   Termchar$B_Class  = [0,0,8,0], !
;   2418  0   Termchar$B_Type  = [1,0,8,0], !
;   2419  0   Termchar$W_Page_Width  = [2,0,16,0], !
;   2420  0   Termchar$3W_Term_Char  = [4,0,24,0], !
;   2421  0   Termchar$B_Page_Length  = [7,0,8,0] !
;   2422  0      TES;
;   2423  0
;   2424  0 MACRO
; M 2425  0  $CRD_Termchar_ATTR =
; M 2426  0   BLOCK [CRD$K_Termchar_Size, BYTE]
; M 2427  0   FIELD ($CRD_Termchar_FLD)
;   2428  0  %;









;     COMMAND QUALIFIERS

; BLISS CRD.R32/LIBRARY

; Run Time:    00:12.1
; Elapsed Time:    00:20.7
; Lines/CPU Min:   12079
; Lexemes/CPU-Min: 45338
; Memory Used: 164 pages
; Library Precompilation Complete
```

```
;    0001   0 %TITLE '*** CRDACT1 Module'
;    0002   0 MODULE CRDACT1 (    ! Error handling action routines
;    0003   0   IDENT = '01-22'
;    0004   0   ) =
;    0005   0 !++
;    0006   0 ! COPYRIGHT (c) 1980, 1981
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! This module contains the routines necessary to perform actions based upon
;    0032  0 ! a DS typecode and a current 'state'. A state table is used to determine what
;    0033  0 ! action routine should be undertaken.
;    0034  0 !
;    0035  0 ! Author:
;    0036  0 !
;    0037  0 ! Jack Stansbury
;    0038  0 !
;    0039  0 ! Creation Date:
;    0040  0 !
;    0041  0 ! 23-April-1982
;    0042  0 !
;    0043  0 ! Version:
;    0044  0 !
;    0045  0 ! 6.9
```

```
;    0046  0 ! Modified By:
;    0047  0 !
;    0048  0 ! 01 Jack Stansbury, Version 1.0, June 21, 1982 (Summer!!!)
;    0049  0 !  Changed CRD$F_DDB to CRD$V_DDB. Also added functionality of the
;    0050  0 !  CRD$V_DDB_Status_End_Testing bit. This bit indicates that everything
;    0051  0 !  has been done on that device.
;    0052  0 !
;    0053  0 ! 02 Jack Stansbury, Version 1.0, June 22, 1982
;    0054  0 !  Took out the Error_Count variables. These are not needed for this version.
;    0055  0 !  Also added some functionality to the Internal_Error routine to allow it
;    0056  0 !  to be used as a general purpose internal error routine (i.e., callable from
;    0057  0 !  anywhere in CRD-AutoTest).
;    0058  0 !
;    0059  0 ! 03 Jack Stansbury, Version 1.0, July 2, 1982
;    0060  0 !  Made changes to the Internal_Error routine so that it will not overwrite
;    0061  0 !  the dispatch vectors before the vectors are no longer needed (i.e., CRD$Stop
;    0062  0 !  needs at least the CRD$Print address in the vectors to print things). Use
;    0063  0 !  a new global vector to store the things in that will later be transferred
;    0064  0 !  to the dispatch area by CRD$Stop.
;    0065  0 !
;    0066  0 ! 04 Jack Stansbury, Version 1.0, July 7, 1982
;    0067  0 !  Added code in the CRD$Preparation_Error routine that will do special case
;    0068  0 !  code for printing the preparation error message. This was necessary because
;    0069  0 !  of the way the message was printed when the user_error text was one character
;    0070  0 !  less in length than the prep_error text. This caused an asterisk to be left
;    0071  0 !  out. I'm not sure if this will fix the problem, but it may help.
;    0072  0 !
;    0073  0 ! 05 Jack Stansbury, Version 1.0, July 9, 1982
;    0074  0 !  Decrement the positive CRD version number when there is an internal error.
;    0075  0 !  This is contrary to the documentation in the DSV$Exit routine because the
;    0076  0 !  code in the routine is wrong (i.e., it does an INCL when it should have done
;    0077  0 !  a DECL). This will have to be changed in the DSV$Exit routine, and in the
;    0078  0 !  CRD$Internal_Error routine later - after CRD-AutoTest goes out.
;    0079  0 !
;    0080  0 ! 06 Jack Stansbury, Version 1.0, July 9, 1982
;    0081  0 !  Make changes to the CRD$Diagnostic_Error routine. It was causing a problem because
;    0082  0 !  I had made a change in the state_table (making a Script_Prompt typecode activate
;    0083  0 !  the Diagnostic_Error routine). The SELECTONE statement in the Internal_Error routine
;    0084  0 !  did not know this new typecode, and it reported an internal error. So, I took out
;    0085  0 !  the SELECTONE statement because it is really not needed. It was in there to keep
;    0086  0 !  track of the number of different types of errors that occurred. I don't do that
;    0087  0 !  anymore.
;    0088  0 !
;    0089  0 ! 07 Jack Stansbury, Version 1.0, July 9, 1982
;    0090  0 !  Make some changes in the Start_Error and Load_Error routines that will make the ending
;    0091  0 !  message come out right. When one start error occurs, the ending message should be the
;    0092  0 !  check device preparation message. To get this to come out, I set the Status_Essential_Dev
;    0093  0 !  bit equal to True. That way, the Exit_CRD routine will think that an essential device
;    0094  0 !  wasn't tested, and print the right message.
;    0095  0 !
;    0096  0 ! 08 Jack Stansbury, Version 1.0, July 9, 1982
;    0097  0 !  Changed the cose in the Preparation_Error routine to handle the special case of the
;    0098  0 !   !#*   FAO construct when the parameter for the  #  is 0. The DS's FAO routine do not
;    0099  0 !  handle this correctly. They eat a character out of the FAO control string. So, I have
;    0100  0 !  to check for this when I print the preparation error messaes, and compensate for it.
```

```
;    0101   0 ! 09 Jack Stansbury, Version 1.1, July 16, 1982
;    0102   0 !  Added functionality for CRD Menu Test.
;    0103   0 !
;    0104   0 ! 10 Jack Stansbury, Version 1.1, July 23, 1982
;    0105   0 !  Fix what is documented in [05] above.
;    0106   0 !
;    0107   0 ! 11 Jack Stansbury, Version 6.9, July 25, 1982
;    0108   0 !  Changed the Preparation error messages back to what they should have been (I fixed the
;    0109   0 !  FAO routines).
;    0110   0 !
;    0111   0 ! 12 Jack Stansbury, Version 1.1, July 25, 1982
;    0112   0 !  Split the CRDACTION module into two different modules: CRDACT1 and CRDACT2. They both
;    0113   0 !  retain the modification history of CRDACTION, but these modification historys become
;    0114   0 !  particular to the module after this point. This module contains the action routines
;    0115   0 !  pertaining to error routines and the global variable declarations.
;    0116   0 !
;    0117   0 ! 13 Jack Stansbury, Version 1.1, August 20, 1982
;    0118   0 !  Changed the DDB status fields.
;    0119   0 !
;    0120   0 ! 14 Jack Stansbury Version 1.1 August 29, 1982
;    0121   0 !  Changed the Device_Okay field to Diag_Passed. Also aborted the diagnostic on a
;    0122   0 !  Preparation_Error.
;    0123   0 !
;    0124   0 ! 15 Jack Stansbury Version 1.1 August 31, 1982
;    0125   0 !  Added the Complete status bit to the DDB status word.
;    0126   0 !
;    0127   0 ! 16 Jack Stansbury Version 1.1 August 31, 1982
;    0128   0 !  If this is Menu Test running, and we are executing the Diagnostic_Error routine,
;    0129   0 !  then we must print 'PASS' or '*FAILED*' since the HS$Advance_Diagnostic routine won't
;    0130   0 !  be executing when a diagnostic finds a diagnostic_error.
;    0131   0 !
;    0132   0 ! 17 Jack Stansbury Version 1.1 September 1, 1982
;    0133   0 !  Took out the above change, so that Menu Test can have its own Diagnostic_Error
;    0134   0 !  code. There were too many differences to share the code. Also, took out the
;    0135   0 !  Error_Count things again! They are not needed for Menu Test either!
;    0136   0 !
;    0137   0 ! 18 Jack Stansbury Version 1.1 September 22, 1982
;    0138   0 !  Changed the definitions of the status bits in the DDBs (again). Also made changes
;    0139   0 !  to the CRD$Stop reasons.
;    0140   0 !
;    0141   0 ! 18 Jack Stansbury Version 1.1 September 30, 1982
;    0142   0 !  Changed some things around for the new state and action routine numbers. Took out CRD$Abort_CRD -
;    0143   0 !  put its functions into CRD$AutoSizer_Error.
;    0144   0 !
;    0145   0 ! 19 Jack Stansbury, Version 1.1, October 1, 1982
;    0146   0 !  Put a $DS_Abort into the AutoSizer_Error routine.
;    0147   0 !
;    0148   0 ! 20 Jack Stansbury, Version 1.1, October 4, 1982
;    0149   0 !  ???
;    0150   0 !
;    0151   0 ! 21 Jack Stansbury, Version 1.1, October 8, 1982
;    0152   0 !  Changed the Exit_CRD routine around again! Hopefully, the last time.
;    0153   0 !
;    0154   0 ! 22 John Ciukaj Version 1.2 15-April-1983
;    0155   0 !  - Changed 'AUTO TEST INCOMPLETE   NO ERRORS' message
```

```
;    0156  0 !     replacing the 'NO ERRORS' text
;    0157  0 !   - Changed ERRPREP message in CRD$Preparation_Error
;    0158  0 !     so that the text is not centered.  Text lines more
;    0159  0 !     than 80 characters  will be more readable.
;    0160  0 !   - Revised algorithm in CRD$Exit_CRD for the completion
;    0161  0 !     message
;    0162  0 !
;    0163  0 ! 23 Scott Cohen Version 1.2 26-Jul-1983
;    0164  0 !   - Changed Preparaton Error Message FAO string
;    0165  0 !--
```

```
;    0166  0 %SBTTL 'Libraries'
;    0167  1 BEGIN     ! Begin the CRDACT1 module
;    0168  1
;    0169  1 LIBRARY
;    0170  1  '$CRD';
;    0171  1
;    0172  1 LIBRARY
;    0173  1  '$DS';
;    0174  1
;    0175  1 LIBRARY
;    0176  1  '$Diag';
;    0177  1
;    0178  1 LIBRARY
;    0179  1  'Sys$Library:Lib';
```

```
;    0180  1 %SBTTL 'Table of Contents'
;    0181  1
;    0182  1 FORWARD ROUTINE     ! In alphabetical order
;    0183  1  CRD$AutoSizer_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    0184  1      ! The AutoSizer had a problem.
;    0185  1
;    0186  1  CRD$Diagnostic_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    0187  1      ! A diagnostic reported an ERRxxxx.
;    0188  1
;    0189  1  CRD$Exit_CRD   : ADDRESSING_MODE (LONG_RELATIVE),
;    0190  1      ! Exit the CRD process
;    0191  1
;    0192  1  CRD$Internal_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    0193  1      ! An internal error (i.e., ERRSUP or CRD) occurred.
;    0194  1
;    0195  1  CRD$Load_Error   : ADDRESSING_MODE (LONG_RELATIVE),
;    0196  1      ! An error occurred while loading a diagnostic
;    0197  1
;    0198  1  CRD$Preparation_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    0199  1      ! An ERRPREP occurred while running a diagnostic.
;    0200  1
;    0201  1  CRD$Print_DDB   : ADDRESSING_MODE (LONG_RELATIVE),
;    0202  1      ! Print contents of a DDB
;    0203  1
;    0204  1  CRD$Start_Error   : ADDRESSING_MODE (LONG_RELATIVE);
;    0205  1      ! An error occurred starting a diagnostic
```

```
;    0206  1 %SBTTL 'Included Macros and Literals'
;    0207  1
;    0208  1  $CRD_Literals;   ! Define some literals for CRD
```

```
;    0209  1 %SBTTL 'Psect Definitions'
;    0210  1
;    0211  1 PSECT
;    0212  1     PLIT   = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0213  1
;    0214  1 PSECT
;    0215  1     CODE   = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0216  1
;    0217  1 PSECT
;    0218  1     OWN    = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0219  1
;    0220  1 PSECT
;    0221  1     GLOBAL = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0222  1 %SBTTL 'Module-Global Static Storage'
;    0223  1
;    0224  1  ModNam ('CRDACT1');  ! Module name for ErrSup macro
```

```
;    0225  1 %SBTTL 'CRD$AutoSizer_Error'
;    0226  1
;    0227  1 GLOBAL ROUTINE CRD$AutoSizer_Error =
;    0228  1
;    0229  1 !++
;    0230  1 ! Functional Description:
;    0231  1 !
;    0232  1 ! An error was found in the AutoSizer (EVSBA). Something went wrong with loading it,
;    0233  1 ! starting it, or in the execution of it. Since we cannot go on with CRD-AutoTest because
;    0234  1 ! none of the devices are attached, we must abort CRD here. So, print a message saying that
;    0235  1 ! CRD-AutoTest is incomplete and exit.
;    0236  1 !
;    0237  1 ! Formal Parameters:
;    0238  1 !
;    0239  1 ! None
;    0240  1 !
;    0241  1 ! Side Effects:
;    0242  1 !
;    0243  1 ! None
;    0244  1 !
;    0245  1 ! Completion Codes:
;    0246  1 !
;    0247  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0248  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0249  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0250  1 !--
```

```
;    0251  2 BEGIN     ! Routine CRD$AutoSizer_Error
;    0252  2  !+
;    0253  2  ! NOTE: This should be fixed so that when an error is detected during AutoSizer processing, this routine   [19]
;    0254  2  ! is called. This rouitne should do the $CRD_Print, and then do a $DS_Abort. The state table should have   [19]
;    0255  2  ! each error condition under the four AutoSizer states be State_AutoSizer_Error/AutoSizer_Error. That way, [19]
;    0256  2  ! this would be called with the typecode = the error typecode, this would abort, causing the next typecode [19]
;    0257  2  ! to be DS_Prompt, which would be the only typecode that the AutoSizer_Error state would have to handle.   [19]
;    0258  2  ! At AutoSizer_Error, DS_Prompt, the state table entry should be State_Start/AutoSizer_Abort. The    [19]
;    0259  2  ! AutoSizer_Abort action routine (new) would call CRD$Stop with the stop reason used below).     [19]
;    0260  2  ! This way, we could abort the AutoSizer and have it clean up it's mess.       [19]
;    0261  2  !-
;    0262  2
;    0263  2  $CRD_Print (CRD$GT_AutoSizer_Error);
;    0264  2      ! Print a message informing the user that something is wrong with the
;    0265  2      ! AutoSizer.
;    0266  2
;    0267  2  CRD$Stop (AUTO$K_Exit_AutoSizer_Error);         ![19]
;    0268  2      ! Exit CRD AutoTest       ![19]
;    0269  2
;    0270  3  RETURN (CRD$K_Success)  ! Do this for BLISS's sake - the above will not return to here  ![19]
;    0271  1 END;     ! Routine CRD$AutoSizer_Error


          .TITLE  CRDACT1 *** CRDACT1 Module
          .IDENT  \01-22\

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      31  54  43  41  44  52  43  07  00000 P.AAA:  .ASCII  <7>\CRDACT1\          ;

          $MODULE=       P.AAA
           .EXTRN  CRD$AUTOSIZER_ERROR
           .EXTRN  CRD$DIAGNOSTIC_ERROR
           .EXTRN  CRD$EXIT_CRD, CRD$INTERNAL_ERROR
           .EXTRN  CRD$LOAD_ERROR, CRD$PREPARATION_ERROR
           .EXTRN  CRD$PRINT_DDB, CRD$START_ERROR
           .EXTRN  CRD$GT_AUTOSIZER_ERROR
           .EXTRN  CRD$PRINT, CRD$STOP

           .PSECT  CODE,NOWRT,  SHR,  PIC,2

             0000 00000     .ENTRY  CRD$AUTOSIZER_ERROR, Save nothing       ; 0227
     00000000G  EF  9F 00002     PUSHAB  CRD$GT_AUTOSIZER_ERROR        ; 0263
         01  DD 00008     PUSHL   #1            ;
         1A  DD 0000A     PUSHL   #26           ;
     00000000G  FF       03  FB 0000C     CALLS   #3, @CRD$PRINT          ;
         02  DD 00013     PUSHL   #2            ; 0267
     00000000G  EF       01  FB 00015     CALLS   #1, CRD$STOP            ;
         50       01  D0 0001C     MOVL    #1, R0           ; 0270
          04 0001F     RET                 ; 0271

; Routine Size:  32 bytes,    Routine Base:  CODE + 0000
```

```
;    0272  1 %SBTTL 'CRD$Diagnostic_Error'
;    0273  1
;    0274  1 GLOBAL ROUTINE CRD$Diagnostic_Error (TypeCode, Error_Message_Length, Error_Message_Address) =
;    0275  1
;    0276  1 !++
;    0277  1 ! Functional Description:
;    0278  1 !
;    0279  1 ! A diagnostic ended in error (i.e., it executed a $DS_ERRxxxx macro).      ![02]
;    0280  1 ! Abort the currently running diagnostic and advance to the next diagnostic.    ![02]
;    0281  1 !
;    0282  1 ! Formal Parameters:
;    0283  1 !
;    0284  1 ! TypeCode       : The typecode of the ERRxxx.
;    0285  1 ! Error_Message_Length  : The length of the error message text.
;    0286  1 ! Error_Message_Address : The address of the error message text.
;    0287  1 !
;    0288  1 ! Side Effects:
;    0289  1 !
;    0290  1 ! Aborts the currently running diagnostic and starts the next one.
;    0291  1 !
;    0292  1 ! Completion Codes:
;    0293  1 !
;    0294  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0295  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0296  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0297  1 !--
```

```
;    0298  2 BEGIN     ! Routine CRD$Diagnostic_Error
;    0299  2  MAP
;    0300  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0301  2
;    0302  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Diagnostic_Error> = True;   ![19]
;    0303  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Device_Bad> = True;    ![18]
;    0304  2      ! Set flag to indicate the device is bad    ![18]
;    0305  2
;    0306  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete> = False;   ![18]
;    0307  2      ! Set flag to indicate that the diagnostic did not complete testing ![18]
;    0308  2
;    0309  2  $CRD_Print (CRD$GT_Fail); ! This device failed testing      ![19]
;    0310  2
;    0311  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed> = True;   ![19]
;    0312  2
;    0313  2  CRD$Print_DDB (.CRD$GA_Current_Diagnostic);
;    0314  2
;    0315  2  IF (.CRD$GB_CRD_Debug) THEN
;    0316  2   $CRD_Print ($ASCIC ('!/CRD$Diagnostic_Error: Aborting the diagnostic.!/'));
;    0317  2
;    0318  2  $DS_Abort (Arg = Program); ! Abort the diagnostic that is running. This will not return to here!
;    0319  2
;    0320  2  RETURN (CRD$K_Success);  ! Do this for BLISS
;    0321  2
;    0322  1 END;     ! Routine CRD$Diagnostic_Error


          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

74  73  6F  6E  67  61  69  44  24  44  52  43  2F  21  32  00008 P.AAB:  .ASCII  \2!/CRD$Diagnostic_Error: Aborting the di\  ;
74  72  6F  62  41  20  3A  72  6F  72  72  45  5F  63  69  00017             ;
      69  64  20  65  68  74  20  67  6E  69  00026             ;
  2F  21  2E  63  69  74  73  6F  6E  67  61  00030    .ASCII  \agnostic.!/\          ;

          .EXTRN  CRD$GA_CURRENT_DIAGNOSTIC
          .EXTRN  CRD$GT_FAIL, CRD$GB_CRD_DEBUG
          .EXTRN  DS$ABORT

          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            0000 00000    .ENTRY  CRD$DIAGNOSTIC_ERROR, Save nothing       ; 0274
          50 00000000G  EF  D0 00002    MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0302
      14  A0       11  88 00009    BISB2   #17, 20(R0)           ; 0303
      14  A0       20  8A 0000D    BICB2   #32, 20(R0)           ; 0306
       00000000G  EF  9F 00011    PUSHAB  CRD$GT_FAIL          ; 0309
          01  DD 00017    PUSHL   #1            ;
          1A  DD 00019    PUSHL   #26           ;
       00000000G  FF       03  FB 0001B    CALLS   #3, @CRD$PRINT          ;
          50 00000000G  EF  D0 00022    MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0311
      14  A0 40    8F  88 00029    BISB2   #64, 20(R0)           ;
          50  DD 0002E    PUSHL   R0            ; 0313
       00000000V  EF       01  FB 00030    CALLS   #1, CRD$PRINT_DDB         ;
          11 00000000G  EF  E9 00037    BLBC    CRD$GB_CRD_DEBUG, 1$         ; 0315
       00000000'  EF  9F 0003E    PUSHAB  P.AAB           ; 0316
```

```
            01  DD 00044    PUSHL   #1              ;
            1A  DD 00046    PUSHL   #26             ;
       00000000G  FF       03  FB 00048    CALLS   #3, @CRD$PRINT          ;
       00000000G  9F       00  FB 0004F 1$:  CALLS   #0, @#DS$ABORT           ; 0318
            50      01  D0 00056    MOVL    #1, R0              ; 0320
         04 00059    RET                 ; 0322

; Routine Size:  90 bytes,    Routine Base:  CODE + 0020
```

```
;    0323  1 %SBTTL 'CRD$Exit_CRD'
;    0324  1
;    0325  1 GLOBAL ROUTINE CRD$Exit_CRD =
;    0326  1
;    0327  1 !++
;    0328  1 ! Functional Description:
;    0329  1 !
;    0330  1 ! Exit the CRD-AutoTest because we are all done executing the diagnostics.
;    0331  1 !
;    0332  1 ! Formal Parameters:
;    0333  1 !
;    0334  1 ! None
;    0335  1 !
;    0336  1 ! Side Effects:
;    0337  1 !
;    0338  1 ! None
;    0339  1 !
;    0340  1 ! Completion Codes:
;    0341  1 !
;    0342  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0343  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0344  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0345  1 !--
```

```
;   0346  2 BEGIN     ! Routine CRD$Exit_CRD
;   0347  2  MAP
;   0348  2   CRD$GA_First_Diagnostic : REF Device_Data_Block_Structure;
;   0349  2
;   0350  2  REGISTER
;   0351  2   Complete  : BYTE INITIAL (True),       ![22]
;   0352  2   Bad_Device  : BYTE INITIAL (False),      ![22]
;   0353  2   Incorrect_Prep  : BYTE INITIAL (False),      ![22]
;   0354  2
;   0355  2   DDB_Ptr    : REF Device_Data_Block_Structure;     ![13]
;   0356  2
;   0357  2  DDB_Ptr = .CRD$GA_First_Diagnostic;
;   0358  2     ! Set DDB_Ptr to point to the first DDB
;   0359  2
;   0360  2  DO
;   0361  3   BEGIN
;   0362  3   REGISTER           ![21]
;   0363  3    Prep_Error  : BYTE,        ![21]
;   0364  3    Bad    : BYTE,         ![22]
;   0365  3    Essential_Device : BYTE,        ![21]
;   0366  3    Finished  : BYTE;        ![21]
;   0367  3
;   0368  3   !+
;   0369  3   ! Step through all the DDB's and accumulate the results of the testing.
;   0370  3   !-
;   0371  3
;   0372  3   CRD$Print_DDB (.DDB_Ptr);
;   0373  3
;   0374  3   Finished  = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete>;  ![21]
;   0375  3   Essential_Device = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device>;   ![21]
;   0376  3   Bad    = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Device_Bad>;   ![22]
;   0377  3   Prep_Error  = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Preparation_Error>;  ![21]
;   0378  3
;   0379  3   !+
;   0380  3   ! Determine if testing is Complete        ![22]
;   0381  3   !-
;   0382  3
;   0383  3   Complete = (.Complete AND .Finished) OR
;   0384  3       (.Complete AND (NOT(.Finished)) AND (NOT(.Essential_Device)));   ![22]
;   0385  3
;   0386  3
;   0387  3   !+
;   0388  3   ! Determine if BAD device          ![22]
;   0389  3   !-
;   0390  3
;   0391  3   Bad_Device = .Bad_Device OR .Bad;        ![22]
;   0392  3
;   0393  3   !+
;   0394  3   ! Determine if Incorrect Preparation        ![22]
;   0395  3   !-
;   0396  3
;   0397  3   Incorrect_Prep = .Incorrect_Prep OR         ![22]
;   0398  4    (            ![21]
;   0399  5     Boolean (.Essential_Device)       ![21]
;   0400  4    AND           ![21]
```

```
;    0401  5     Boolean (.Prep_Error)        ![21]
;    0402  3    );            ![21]
;    0403  3
;    0404  3    !+
;    0405  3    ! Advance to the next DDB in the queue.
;    0406  3    !-
;    0407  3
;    0408  3    DDB_Ptr = .DDB_Ptr [CRD$K_DDB_FLink];        ![13]
;    0409  3    END
;    0410  2   UNTIL (.DDB_Ptr EQL .CRD$GA_First_Diagnostic);
;    0411  2
;    0412  2   IF (.CRD$GB_CRD_Debug) THEN
; P  0413  2    $CRD_Print ($ASCIC ('Exit_CRD: Complete    = !XB!/',      ![22]
; P  0414  2              '           Incorrect_Prep      = !XB!/',      ![22]
; P  0415  2              '           Bad_Device = !XB!/'),      ![22]
;    0416  2     .Complete, .Incorrect_Prep, .Bad_Device);      ![22]
;    0417  2
;    0418  2   !+
;    0419  2   ! Check the flags to see if we found anything.         ![13]
;    0420  2   !-
;    0421  2
;    0422  2   IF (.Bad_Device) THEN           ![22]
;    0423  2
;    0424  2    CRD$Stop (AUTO$K_Exit_Errors_InComplete)        ![22]
;    0425  2      ! AUTOTEST INCOMPLETE - XXX BAD - CALL FIELD SERVICE    ![22]
;    0426  2
;    0427  2   ELSE IF (.Incorrect_Prep) THEN          ![22]
;    0428  2
;    0429  2    CRD$Stop (AUTO$K_Exit_NoErrors_Prep)        ![21]
;    0430  2      ! AUTOTEST INCOMPLETE        ![22]
;    0431  2      ! CHECK DEVICE PREPARATION - IF OK, THEN CALL FIELD SERVICE   ![22]
;    0432  2
;    0433  2   ELSE IF (NOT(.Complete)) THEN          ![22]
;    0434  2
;    0435  2    CRD$Stop (AUTO$K_Exit_NoErrors_Incomplete)       ![21]
;    0436  2      ! AUTOTEST INCOMPLETE - IF ASSISTANCE NEEDED, CALL FIELD SERVICE  ![22]
;    0437  2
;    0438  2   ELSE              ![21]
;    0439  2
;    0440  2    CRD$Stop (AUTO$K_Exit_NoErrors_Complete);       ![21]
;    0441  2      ! AUTOTEST COMPLETE - NO ERRORS       ![22]
;    0442  2
;    0443  2   !+
;    0444  2   ! None of the above calls to CRD$Stop will return here.
;    0445  2   !-
;    0446  2
;    0447  3    RETURN (CRD$K_Success)  ! Do this for BLISS's sake
;    0448  1  END;     ! Routine CRD$Exit_CRD


            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

70  6D  6F  43  20  3A  44  52  43  5F  74  69  78  45  5F  0003B P.AAC:  .ASCII  \_Exit_CRD: Complete    = !XB!/          \  ;
2F  21  42  58  21  20  3D  20  20  20  20  65  74  65  6C  0004A             ;
```

```
      20  20  20  20  20  20  20  20  20  20  00059                ;
20  70  65  72  50  5F  74  63  65  72  72  6F  63  6E  49  00063    .ASCII  \Incorrect_Prep       = !XB!/          Ba\  ;
20  20  2F  21  42  58  21  20  3D  20  20  20  20  20  20  00072                ;
      61  42  20  20  20  20  20  20  20  20  00081                ;
21  42  58  21  20  3D  20  65  63  69  76  65  44  5F  64  0008B    .ASCII  \d_Device = !XB!/\         ;
      2F  0009A                ;

          .EXTRN  CRD$GA_FIRST_DIAGNOSTIC

          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            07FC 00000    .ENTRY  CRD$EXIT_CRD, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 0325
           R10                ;
          52       01  90 00002    MOVB    #1, COMPLETE          ; 0346
          53  94 00005    CLRB    BAD_DEVICE          ;
          54  94 00007    CLRB    INCORRECT_PREP          ;
          55 00000000G  EF  D0 00009    MOVL    CRD$GA_FIRST_DIAGNOSTIC, DDB_PTR       ; 0357
          55  DD 00010 1$:   PUSHL   DDB_PTR          ; 0372
    00000000V  EF       01  FB 00012    CALLS   #1, CRD$PRINT_DDB          ;
    50      14   A5         01      05  EF 00019    EXTZV   #5, #1, 20(DDB_PTR), R0        ; 0374
          57      50  90 0001F    MOVB    R0, FINISHED          ;
    50      14   A5         01      02  EF 00022    EXTZV   #2, #1, 20(DDB_PTR), R0        ; 0375
          56      50  90 00028    MOVB    R0, ESSENTIAL_DEVICE          ;
    50      14   A5         01      04  EF 0002B    EXTZV   #4, #1, 20(DDB_PTR), R0        ; 0376
          51      50  90 00031    MOVB    R0, BAD          ;
    58      14   A5         01      01  EF 00034    EXTZV   #1, #1, 20(DDB_PTR), R8        ; 0377
          50      58  90 0003A    MOVB    R8, PREP_ERROR          ;
          58      52  9A 0003D    MOVZBL  COMPLETE, R8          ; 0383
          59      57  9A 00040    MOVZBL  FINISHED, R9          ;
          59      59  D2 00043    MCOML   R9, R9          ;
          58      59  CA 00046    BICL2   R9, R8          ;
          59      52  9A 00049    MOVZBL  COMPLETE, R9          ; 0384
          57      57  9A 0004C    MOVZBL  FINISHED, R7          ;
          57        59      57  CB 0004F    BICL3   R7, R9, R7          ;
          5A      56  9A 00053    MOVZBL  ESSENTIAL_DEVICE, R10          ;
          57      5A  CA 00056    BICL2   R10, R7          ;
          52        57      58  89 00059    BISB3   R8, R7, COMPLETE          ;
          53      51  88 0005D    BISB2   BAD, BAD_DEVICE          ; 0391
          51      56  9A 00060    MOVZBL  ESSENTIAL_DEVICE, R1          ; 0401
          50      50  9A 00063    MOVZBL  PREP_ERROR, R0          ;
          50      50  D2 00066    MCOML   R0, R0          ;
          50        51      50  CB 00069    BICL3   R0, R1, R0          ;
57          50        01       00  EF 0006D    EXTZV   #0, #1, R0, R7          ; 0398
          54      57  88 00072    BISB2   R7, INCORRECT_PREP          ;
          55      65  D0 00075    MOVL    (DDB_PTR), DDB_PTR          ; 0408
    00000000G  EF       55  D1 00078    CMPL    DDB_PTR, CRD$GA_FIRST_DIAGNOSTIC       ; 0410
          8F  12 0007F    BNEQ    1$          ;
          1A 00000000G  EF  E9 00081    BLBC    CRD$GB_CRD_DEBUG, 2$          ; 0412
          7E      53  9A 00088    MOVZBL  BAD_DEVICE, -(SP)          ; 0416
          7E      54  9A 0008B    MOVZBL  INCORRECT_PREP, -(SP)          ;
          7E      52  9A 0008E    MOVZBL  COMPLETE, -(SP)          ;
    00000000'  EF  9F 00091    PUSHAB  P.AAC          ;
          01  DD 00097    PUSHL   #1          ;
          1A  DD 00099    PUSHL   #26          ;
    00000000G  FF       06  FB 0009B    CALLS   #6, @CRD$PRINT          ;
```

```
            04      53  E9 000A2 2$:   BLBC    BAD_DEVICE, 3$          ; 0422
            04  DD 000A5    PUSHL   #4              ; 0424
            10  11 000A7    BRB     6$              ;
            04      54  E9 000A9 3$:   BLBC    INCORRECT_PREP, 4$          ; 0427
            07  DD 000AC    PUSHL   #7              ; 0429
            09  11 000AE    BRB     6$              ;
            04      52  E8 000B0 4$:   BLBS    COMPLETE, 5$            ; 0433
            06  DD 000B3    PUSHL   #6              ; 0435
            02  11 000B5    BRB     6$              ;
            05  DD 000B7 5$:   PUSHL   #5              ; 0440
      00000000G  EF      01  FB 000B9 6$:   CALLS   #1, CRD$STOP           ;
            50      01  D0 000C0    MOVL    #1, R0              ; 0447
        04 000C3    RET                 ; 0448

; Routine Size:  196 bytes,    Routine Base:  CODE + 007A
```

```
;    0449  1 %SBTTL 'CRD$Internal_Error'
;    0450  1
;    0451  1 GLOBAL ROUTINE CRD$Internal_Error (TypeCode, Length, Address) =
;    0452  1
;    0453  1 !++
;    0454  1 ! Functional Description:
;    0455  1 !
;    0456  1 ! An internal error (i.e., a software bug) was detected in CRD. Exit CRD.
;    0457  1 !
;    0458  1 ! Formal Parameters:
;    0459  1 !
;    0460  1 ! TypeCode: The typecode that was in effect when the internal error occurred.
;    0461  1 ! Address:  The address of what the Resident-DS was trying to print.
;    0462  1 ! Length:   The length of . . . .
;    0463  1 !
;    0464  1 ! Side Effects:
;    0465  1 !
;    0466  1 ! None
;    0467  1 !
;    0468  1 ! Completion Codes:
;    0469  1 !
;    0470  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0471  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0472  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0473  1 !--
```

```
;    0474  2 BEGIN     ! Routine CRD$Internal_Error
;    0475  2  MAP
;    0476  2   CRD$AL_CRD_Dispatch_Vectors : VECTOR [4, BYTE];
;    0477  2       ! Reference the dispatch vector information as an array of bytes
;    0478  2       ! Do not worry about these being SIGNED bytes because only the
;    0479  2       ! positive version number is referenced.
;    0480  2
;    0481  2   !+
;    0482  2   ! Store pertinent information about the state of CRD, and call the CRD$Stop routine to stop CRD!
;    0483  2   !-
;    0484  2
;    0485  2   CRD$AL_CRD_Dispatch_Vectors [1] = .CRD$AL_CRD_Dispatch_Vectors [1] + 1;     ![10]
;    0486  2       ! Increment the positive version number as a cue to the
;    0487  2       ! DSV$Exit routine to allow it to print the following CRD information.
;    0488  2
;    0489  2   CRD$GL_CRD_Debug_Vector [0] = .CRD$GA_Current_Diagnostic;      ![03]
;    0490  2   CRD$GL_CRD_Debug_Vector [1] = .CRD$GA_First_Diagnostic;       ![03]
;    0491  2   CRD$GL_CRD_Debug_Vector [2] = .CRD$GL_Current_TypeCode;       ![03]
;    0492  2   CRD$GL_CRD_Debug_Vector [3] = .CRD$GL_Current_State;       ![03]
;    0493  2   CRD$GL_CRD_Debug_Vector [4] = .CRD$GL_Current_Action;       ![03]
;    0494  2   CRD$GL_CRD_Debug_Vector [5] = .Address;          ![03]
;    0495  2   CRD$GL_CRD_Debug_Vector [6] = .Length;         ![03]
;    0496  2   CRD$GL_CRD_Debug_Vector [7] = .TypeCode;         ![03]
;    0497  2       ! This could be an Internal_Error code.     ![02]
;    0498  2
;    0499  2   IF (.Address NEQ 0) THEN          ![02]
;    0500  2    CRD$GL_CRD_Debug_Vector [8] = ..Address        ![03]
;    0501  2   ELSE
;    0502  2    CRD$GL_CRD_Debug_Vector [8] = %X'FFFFFFFF';       ![03]
;    0503  2       ! Put this in as a clue to show that the address is not indirect ![03]
;    0504  2
;    0505  2   !+              ![02]
;    0506  2   ! This routine MUST call CRD$Stop here because CRD$Internal_Error is called from routines for other  ![03]
;    0507  2   ! than action routine uses.          ![03]
;    0508  2   !-              ![02]
;    0509  2
;    0510  2   CRD$Stop (AUTO$K_Exit_Internal_Error);         ![18]
;    0511  3   RETURN (CRD$K_Success)
;    0512  1 END;     ! Routine CRD$Internal_Error


        .EXTRN  CRD$AL_CRD_DISPATCH_VECTORS
        .EXTRN  CRD$GL_CRD_DEBUG_VECTOR
        .EXTRN  CRD$GL_CURRENT_TYPECODE
        .EXTRN  CRD$GL_CURRENT_STATE
        .EXTRN  CRD$GL_CURRENT_ACTION

          0000 00000     .ENTRY  CRD$INTERNAL_ERROR, Save nothing       ; 0451
00000000G  EF  96 00002     INCB    CRD$AL_CRD_DISPATCH_VECTORS+1        ; 0485
00000000G  EF 00000000G  EF  D0 00008     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, -         ; 0489
     CRD$GL_CRD_DEBUG_VECTOR         ;
00000000G  EF 00000000G  EF  D0 00013     MOVL    CRD$GA_FIRST_DIAGNOSTIC, -         ; 0490
     CRD$GL_CRD_DEBUG_VECTOR+4       ;
00000000G  EF 00000000G  EF  D0 0001E     MOVL    CRD$GL_CURRENT_TYPECODE, -         ; 0491
```

```
                CRD$GL_CRD_DEBUG_VECTOR+8       ;
        00000000G  EF 00000000G  EF  D0 00029    MOVL    CRD$GL_CURRENT_STATE, -         ; 0492
                CRD$GL_CRD_DEBUG_VECTOR+12      ;
        00000000G  EF 00000000G  EF  D0 00034    MOVL    CRD$GL_CURRENT_ACTION, -        ; 0493
                CRD$GL_CRD_DEBUG_VECTOR+16      ;
        00000000G  EF 0C    AC  D0 0003F    MOVL    ADDRESS, CRD$GL_CRD_DEBUG_VECTOR+20     ; 0494
        00000000G  EF 08    AC  D0 00047    MOVL    LENGTH, CRD$GL_CRD_DEBUG_VECTOR+24      ; 0495
        00000000G  EF 04    AC  D0 0004F    MOVL    TYPECODE, CRD$GL_CRD_DEBUG_VECTOR+28    ; 0496
      0C    AC  D5 00057    TSTL    ADDRESS         ; 0499
            0A  13 0005A    BEQL    1$              ;
        00000000G  EF 0C    BC  D0 0005C    MOVL    @ADDRESS, CRD$GL_CRD_DEBUG_VECTOR+32    ; 0500
            07  11 00064    BRB     2$              ;
        00000000G  EF       01  CE 00066 1$:    MNEGL   #1, CRD$GL_CRD_DEBUG_VECTOR+32      ; 0502
            01  DD 0006D 2$:    PUSHL   #1          ; 0510
        00000000G  EF       01  FB 0006F    CALLS   #1, CRD$STOP            ;
            50      01  D0 00076    MOVL    #1, R0          ; 0511
             04 00079    RET             ; 0512

; Routine Size:  122 bytes,    Routine Base:  CODE + 013E
```

```
;    0513  1 %SBTTL 'CRD$Load_Error'
;    0514  1
;    0515  1 GLOBAL ROUTINE CRD$Load_Error =
;    0516  1
;    0517  1 !++
;    0518  1 ! Functional Description:
;    0519  1 !
;    0520  1 ! There was an error in loading the diagnostic. Print a message telling the operator
;    0521  1 ! this and go to the next diagnostic.
;    0522  1 !
;    0523  1 ! Formal Parameters:
;    0524  1 !
;    0525  1 ! None
;    0526  1 !
;    0527  1 ! Side Effects:
;    0528  1 !
;    0529  1 ! None
;    0530  1 !
;    0531  1 ! Completion Codes:
;    0532  1 !
;    0533  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0534  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0535  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0536  1 !--
```

```
;    0537  2 BEGIN     ! Routine CRD$Load_Error
;    0538  2  MAP
;    0539  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0540  2
;    0541  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed> = True;   ![19]
;    0542  2      ! We will print a message here      ![19]
;    0543  2
;    0544  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete> = False;   ![18]
;    0545  2      ! We did not complete testing on this device.    ![18]
;    0546  2
;    0547  2  !+
;    0548  2  ! Print a message indicating that the diagnostic could not be loaded for some reason.
;    0549  2  ! Return Repeat_Turing so that the Turing_Machine can do an Advance_State (to the
;    0550  2  ! Advance_Diagnostic state).
;    0551  2  !-
;    0552  2
;    0553  2  $CRD_Print (CRD$GT_Cannot_Load_Diag);
;    0554  2
;    0555  2  CRD$Print_DDB (.CRD$GA_Current_Diagnostic);
;    0556  2
;    0557  3  RETURN (CRD$K_Repeat_Turing)
;    0558  1 END;     ! Routine CRD$Load_Error


          .EXTRN  CRD$GT_CANNOT_LOAD_DIAG

            0000 00000     .ENTRY  CRD$LOAD_ERROR, Save nothing        ; 0515
         50 00000000G   EF   D0 00002     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0541
      14   A0 40    8F   88 00009     BISB2   #64, 20(R0)            ;
      14   A0       20   8A 0000E     BICB2   #32, 20(R0)            ; 0544
       00000000G   EF   9F 00012     PUSHAB  CRD$GT_CANNOT_LOAD_DIAG         ; 0553
            01   DD 00018     PUSHL   #1              ;
            1A   DD 0001A     PUSHL   #26             ;
       00000000G   FF       03   FB 0001C     CALLS   #3, @CRD$PRINT          ;
       00000000G   EF   DD 00023     PUSHL   CRD$GA_CURRENT_DIAGNOSTIC         ; 0555
       00000000V   EF       01   FB 00029     CALLS   #1, CRD$PRINT_DDB          ;
            50       02   D0 00030     MOVL    #2, R0             ; 0557
          04 00033     RET                 ; 0558

; Routine Size:  52 bytes,    Routine Base:  CODE + 01B8
```

```
;    0559  1 %SBTTL 'CRD$Preparation_Error'
;    0560  1
;    0561  1 GLOBAL ROUTINE CRD$Preparation_Error =
;    0562  1
;    0563  1 !++
;    0564  1 ! Functional Description:
;    0565  1 !
;    0566  1 ! A preparation error occurred while running a diagnostic (not EVSBA, the AutoSizer though).
;    0567  1 ! Print a message saying that this test is incomplete and return Repeat_Turing. The state
;    0568  1 ! table will advance to the Advance_Diagnostic state.
;    0569  1 !
;    0570  1 ! Formal Parameters:
;    0571  1 !
;    0572  1 ! None
;    0573  1 !
;    0574  1 ! Side Effects:
;    0575  1 !
;    0576  1 ! None
;    0577  1 !
;    0578  1 ! Completion Codes:
;    0579  1 !
;    0580  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0581  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0582  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0583  1 !--
```

```
;    0584  2 BEGIN     ! Routine CRD$Preparation_Error
;    0585  2  MAP
;    0586  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0587  2
;    0588  2  BIND
;    0589  3   Diag_Vector_Ptr = (.CRD$GA_Current_Diagnostic [CRD$K_DDB_Auto_Support_Entry])
;    0590  2    : Hardware_Support_Vector;
;    0591  2
;    0592  2  BIND
;    0593  2   Prep_Error_Text = (.Diag_Vector_Ptr [CRD$K_Preparation_Error_Text]) : VECTOR [, BYTE];
;    0594  2
;    0595  2  BIND
;    0596  2   User_Error_Text_Address = (.CRD$GA_User_Error_Text)  : VECTOR [, BYTE],
;    0597  2   User_Error_Text  = (..CRD$GA_User_Error_Text) : VECTOR [, BYTE];
;    0598  2
;    0599  2  LOCAL
;    0600  2   No_User_Text : BYTE INITIAL (False);
;    0601  2
;    0602  2  !+
;    0603  2  ! Indicate in the DDB status bits that this device had a prep error and that we did not complete testing.
;    0604  2  !-
;    0605  2
;    0606  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Preparation_Error> = True;   ![13]
;    0607  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed>   = True;   ![19]
;    0608  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete> = False;   ![18]
;    0609  2
;    0610  2  !+
;    0611  2  ! Print a message saying that we got a preparation error. The testing will not be
;    0612  2  ! done on the current device (i.e., since the hardware is not properly setup for the
;    0613  2  ! current diagnostic, the diagnostic cannot be run).
;    0614  2  !-
;    0615  2
;    0616  2  !+
;    0617  2  ! These checks must be made in this order. Even though it appears they could be made into one IF
;    0618  2  ! statement involving an OR, this is not so because BLISS does not guarantee in what order the
;    0619  2  ! expressions will be evaluated.
;    0620  2  !-
;    0621  2
;    0622  2  IF (.User_Error_Text_Address EQL 0) THEN
;    0623  2   No_User_Text = True;
;    0624  2
;    0625  2  IF (NOT .No_User_Text) THEN
;    0626  2   IF (.User_Error_Text [0] EQL 0) THEN
;    0627  2    No_User_Text = True;
;    0628  2
;    0629  2  $CRD_Print (CRD$GT_New_Line); ! Print a <CR><LF>.
;    0630  2
;    0631  2  IF (.No_User_Text) THEN
;    0632  3   BEGIN
;    0633  3   !+
;    0634  3   ! Either the address passed to DSX$ErrPrep is 0, or the user has specified a null ASCIC string.
;    0635  3   ! Do not print the first line (since there is no user error text).
;    0636  3   !-
;    0637  3   $CRD_Print (CRD$GT_Prep_Error_Spaces, Prep_Error_Text [0]);     ![23]
;    0638  3   END
```

```
;    0639  2  ELSE
;    0640  3   BEGIN
;    0641  3   LOCAL              ![04]
;    0642  3    Difference;           ![04]
;    0643  3
;    0644  3   Difference = ABS (.User_Error_Text [0] - .Prep_Error_Text [0]);     ![04]
;    0645  3
;    0646  3   IF (.User_Error_Text [0] GTR .Prep_Error_Text [0]) THEN
;    0647  4    BEGIN
;    0648  4    !+
;    0649  4    ! The first line will be longer than the second line. Thus, the second line must be
;    0650  4    ! centered. The first line needs no spaces.      ![04]
;    0651  4    !-
;    0652  4    LOCAL
;    0653  4     Right, Left;
;    0654  4
;    0655  4    !Left  = (.Difference + 1) / 2;        ![04]
;    0656  4    !Right = .Difference - .Left;        ![04]
;    0657  4    Right=1;
;    0658  4    Left=1;   !        [22]
;    0659  4
;    0660  4    $CRD_Print (CRD$GT_Prep_Error_Spaces,   User_Error_Text [0]);  ![23]
;    0661  4    $CRD_Print (CRD$GT_Prep_Error_Spaces,    Prep_Error_Text [0]);  ![23]
;    0662  4    END
;    0663  3   ELSE
;    0664  4    BEGIN
;    0665  4    !+
;    0666  4    ! The second line will be longer than the first line. Thus, the first line must be
;    0667  4    ! centered. The second line needs no spaces.      ![04]
;    0668  4    !-
;    0669  4    LOCAL
;    0670  4     Right, Left;
;    0671  4
;    0672  4    !Left  = (.Difference + 1) / 2;        ![04]
;    0673  4    !Right = .Difference - .Left;        ![04]
;    0674  4    Left=1;
;    0675  4    Right=1;  !        [22]
;    0676  4    $CRD_Print (CRD$GT_Prep_Error_Spaces,    User_Error_Text [0]);    ![23]
;    0677  4    $CRD_Print (CRD$GT_Prep_Error_Spaces,  Prep_Error_Text [0]);    ![23]
;    0678  4    END
;    0679  2    END;
;    0680  2
;    0681  2   $CRD_Print (CRD$GT_New_Line); ! Print a <CR><LF>.
;    0682  2
;    0683  2   CRD$Print_DDB (.CRD$GA_Current_Diagnostic);
;    0684  2
;    0685  2   $DS_Abort (Arg = Program); ! Abort the currently running diagnostic program   ![14]
;    0686  2
;    0687  2   RETURN (CRD$K_Success);  ! This will not be executed because the above will not return to here ![19]
;    0688  1  END;      ! Routine CRD$Preparation_Error


          .EXTRN  CRD$GA_USER_ERROR_TEXT
          .EXTRN  CRD$GT_NEW_LINE
```

```
                .EXTRN  CRD$GT_PREP_ERROR_SPACES

                  003C 00000     .ENTRY  CRD$PREPARATION_ERROR, Save R2,R3,R4,R5      ; 0561
                50 00000000G   EF  D0 00002     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0589
                52 0C    A0  D0 00009     MOVL    12(R0), R2              ;
                54 00000000G   FF  D0 0000D     MOVL    @CRD$GA_USER_ERROR_TEXT, R4        ; 0597
                55   94 00014     CLRB    NO_USER_TEXT            ;
             14 A0 42    8F  88 00016     BISB2   #66, 20(R0)             ; 0607
             14 A0       20  8A 0001B     BICB2   #32, 20(R0)             ; 0608
              00000000G   FF  D5 0001F     TSTL    @CRD$GA_USER_ERROR_TEXT         ; 0622
                03   12 00025     BNEQ    1$                      ;
                55       01  90 00027     MOVB    #1, NO_USER_TEXT          ; 0623
                07       55  E8 0002A 1$:   BLBS    NO_USER_TEXT, 2$          ; 0625
                64   95 0002D     TSTB    (R4)                    ; 0626
                03   12 0002F     BNEQ    2$                      ;
                55       01  90 00031     MOVB    #1, NO_USER_TEXT          ; 0627
              00000000G   EF  9F 00034 2$:   PUSHAB  CRD$GT_NEW_LINE          ; 0629
                01   DD 0003A     PUSHL   #1                      ;
                1A   DD 0003C     PUSHL   #26                     ;
              00000000G   FF       03  FB 0003E     CALLS   #3, @CRD$PRINT          ;
                53 00000000G   EF  D0 00045     MOVL    CRD$PRINT, R3           ; 0637
                12       55  E9 0004C     BLBC    NO_USER_TEXT, 3$          ; 0631
             1C A2  DD 0004F     PUSHL   28(R2)                  ; 0637
              00000000G   EF  9F 00052     PUSHAB  CRD$GT_PREP_ERROR_SPACES        ;
                01   DD 00058     PUSHL   #1                      ;
                1A   DD 0005A     PUSHL   #26                     ;
                63       04  FB 0005C     CALLS   #4, (R3)            ;
                35   11 0005F     BRB     5$                      ; 0631
                50       64  9A 00061 3$:   MOVZBL  (R4), R0           ; 0644
                51 1C    B2  9A 00064     MOVZBL  @28(R2), R1           ;
                50       51  C2 00068     SUBL2   R1, R0          ;
                03   18 0006B     BGEQ    4$                      ;
                50       50  CE 0006D     MNEGL   DIFFERENCE, DIFFERENCE        ;
                50       01  D0 00070 4$:   MOVL    #1, RIGHT           ; 0675
                54   DD 00073     PUSHL   R4                      ; 0676
              00000000G   EF  9F 00075     PUSHAB  CRD$GT_PREP_ERROR_SPACES        ;
                01   DD 0007B     PUSHL   #1                      ;
                1A   DD 0007D     PUSHL   #26                     ;
                63       04  FB 0007F     CALLS   #4, (R3)            ;
             1C A2  DD 00082     PUSHL   28(R2)                  ; 0677
              00000000G   EF  9F 00085     PUSHAB  CRD$GT_PREP_ERROR_SPACES        ;
                01   DD 0008B     PUSHL   #1                      ;
                1A   DD 0008D     PUSHL   #26                     ;
              00000000G   FF       04  FB 0008F     CALLS   #4, @CRD$PRINT          ;
              00000000G   EF  9F 00096 5$:   PUSHAB  CRD$GT_NEW_LINE          ; 0681
                01   DD 0009C     PUSHL   #1                      ;
                1A   DD 0009E     PUSHL   #26                     ;
              00000000G   FF       03  FB 000A0     CALLS   #3, @CRD$PRINT          ;
              00000000G   EF  DD 000A7     PUSHL   CRD$GA_CURRENT_DIAGNOSTIC         ; 0683
              00000000V   EF       01  FB 000AD     CALLS   #1, CRD$PRINT_DDB           ;
              00000000G   9F       00  FB 000B4     CALLS   #0, @#DS$ABORT          ; 0685
                50       01  D0 000BB     MOVL    #1, R0          ; 0687
                 04 000BE     RET                 ; 0688

; Routine Size:  191 bytes,    Routine Base:  CODE + 01EC
```

```
;    0689  1 %SBTTL 'CRD$Print_DDB'
;    0690  1
;    0691  1 GLOBAL ROUTINE CRD$Print_DDB (DDB_Ptr) =
;    0692  1
;    0693  1 !++
;    0694  1 ! Functional Description:
;    0695  1 !
;    0696  1 ! None
;    0697  1 !
;    0698  1 ! Formal Parameters:
;    0699  1 !
;    0700  1 ! DDB_Ptr: The address of a Device Data Block
;    0701  1 !
;    0702  1 ! Side Effects:
;    0703  1 !
;    0704  1 ! None
;    0705  1 !
;    0706  1 ! Completion Codes:
;    0707  1 !
;    0708  1 ! CRD$K_Success
;    0709  1 !--
```

```
;   0710  2 BEGIN     ! Routine CRD$Print_DDB
;   0711  2  MAP
;   0712  2   DDB_Ptr : REF Device_Data_Block_Structure;
;   0713  2
;   0714  2  IF (.CRD$GB_CRD_Debug) THEN
;   0715  3   BEGIN
;   0716  3   $CRD_Print ($ASCIC ('!/Printing the contents of the !XL DDB:!/'), .DDB_Ptr);
;   0717  3   $CRD_Print ($ASCIC ('  FLink:               !XL!/'), .DDB_Ptr [CRD$K_DDB_FLink]);
;   0718  3   $CRD_Print ($ASCIC ('  BLink:               !XL!/'), .DDB_Ptr [CRD$K_DDB_BLink]);
;   0719  3   $CRD_Print ($ASCIC ('  PTable_Entry:        !XL!/'), .DDB_Ptr [CRD$K_DDB_PTable_Entry]);
;   0720  3   $CRD_Print ($ASCIC ('  Auto_Support_Entry: !XL!/'), .DDB_Ptr [CRD$K_DDB_Auto_Support_Entry]);
;   0721  3   $CRD_Print ($ASCIC ('  Menu_Support_Entry: !XL!/'), .DDB_Ptr [CRD$K_DDB_Menu_Support_Entry]);
;   0722  3   $CRD_Print ($ASCIC ('  Status:              !XL!/'), .DDB_Ptr [CRD$K_DDB_Status]);
;   0723  2   END;
;   0724  2
;   0725  2  RETURN (CRD$K_Success);
;   0726  1 END;     ! Routine CRD$Print_DDB



          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

65  68  74  20  67  6E  69  74  6E  69  72  50  2F  21  29  0009B P.AAD:  .ASCII  \)!/Printing the contents of the !XL DDB:\  ;
68  74  20  66  6F  20  73  74  6E  65  74  6E  6F  63  20  000AA             ;
      3A  42  44  44  20  4C  58  21  20  65  000B9                 ;
          2F  21  000C3    .ASCII  \!/\              ;
20  20  20  20  20  20  3A  6B  6E  69  4C  46  20  20  1B  000C5 P.AAE:  .ASCII  <27>\  FLink:               !XL!/\       ;
 2F  21  4C  58  21  20  20  20  20  20  20  20  20  000D4                 ;
20  20  20  20  20  20  3A  6B  6E  69  4C  42  20  20  1B  000E1 P.AAF:  .ASCII  <27>\  BLink:               !XL!/\       ;
 2F  21  4C  58  21  20  20  20  20  20  20  20  20  000F0                 ;
79  72  74  6E  45  5F  65  6C  62  61  54  50  20  20  1B  000FD P.AAG:  .ASCII  <27>\  PTable_Entry:        !XL!/\       ;
 2F  21  4C  58  21  20  20  20  20  20  20  20  3A  0010C                 ;
74  72  6F  70  70  75  53  5F  6F  74  75  41  20  20  1B  00119 P.AAH:  .ASCII  <27>\  Auto_Support_Entry: !XL!/\       ;
 2F  21  4C  58  21  20  3A  79  72  74  6E  45  5F  00128                 ;
74  72  6F  70  70  75  53  5F  75  6E  65  4D  20  20  1B  00135 P.AAI:  .ASCII  <27>\  Menu_Support_Entry: !XL!/\       ;
 2F  21  4C  58  21  20  3A  79  72  74  6E  45  5F  00144                 ;
20  20  20  20  20  3A  73  75  74  61  74  53  20  20  1B  00151 P.AAJ:  .ASCII  <27>\  Status:              !XL!/\       ;
 2F  21  4C  58  21  20  20  20  20  20  20  20  20  00160                 ;



          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            0004 00000     .ENTRY  CRD$PRINT_DDB, Save R2        ; 0691
          03 00000000G  EF  E8 00002     BLBS    CRD$GB_CRD_DEBUG, 1$          ; 0714
        008E  31 00009     BRW     2$            ;
          52 04   AC  D0 0000C 1$:   MOVL    DDB_PTR, R2           ; 0716
          52  DD 00010     PUSHL   R2            ;
      00000000'  EF  9F 00012     PUSHAB  P.AAD           ;
          01  DD 00018     PUSHL   #1            ;
          1A  DD 0001A     PUSHL   #26           ;
      00000000G  FF       04  FB 0001C     CALLS   #4, aCRD$PRINT        ;
          62  DD 00023     PUSHL   (R2)            ; 0717
      00000000'  EF  9F 00025     PUSHAB  P.AAE           ;
          01  DD 0002B     PUSHL   #1            ;
```

```
                1A  DD 0002D    PUSHL   #26                     ;
          0000v000G  FF         04  FB 0002F    CALLS   #4, aCRD$PRINT          ;
              04    A2  DD 00036    PUSHL   4(R2)                   ; 0718
               00000000'  EF  9F 00039    PUSHAB  P.AAF                   ;
                    01  DD 0003F    PUSHL   #1                  ;
                    1A  DD 00041    PUSHL   #26                  ;
               00000000G  FF         04  FB 00043    CALLS   #4, aCRD$PRINT          ;
              08    A2  DD 0004A    PUSHL   8(R2)                   ; 0719
               00000000'  EF  9F 0004D    PUSHAB  P.AAG                   ;
                    01  DD 00053    PUSHL   #1                  ;
                    1A  DD 00055    PUSHL   #26                  ;
               00000000G  FF         04  FB 00057    CALLS   #4, aCRD$PRINT          ;
              0C    A2  DD 0005E    PUSHL   12(R2)                  ; 0720
               00000000'  EF  9F 00061    PUSHAB  P.AAH                   ;
                    01  DD 00067    PUSHL   #1                  ;
                    1A  DD 00069    PUSHL   #26                  ;
               00000000G  FF         04  FB 0006B    CALLS   #4, aCRD$PRINT          ;
              10    A2  DD 00072    PUSHL   16(R2)                  ; 0721
               00000000'  EF  9F 00075    PUSHAB  P.AAI                   ;
                    01  DD 0007B    PUSHL   #1                  ;
                    1A  DD 0007D    PUSHL   #26                  ;
               00000000G  FF         04  FB 0007F    CALLS   #4, aCRD$PRINT          ;
              14    A2  DD 00086    PUSHL   20(R2)                  ; 0722
               00000000'  EF  9F 00089    PUSHAB  P.AAJ                   ;
                    01  DD 0008F    PUSHL   #1                  ;
                    1A  DD 00091    PUSHL   #26                  ;
               00000000G  FF         04  FB 00093    CALLS   #4, aCRD$PRINT          ;
                    50       01  D0 0009A 2$:   MOVL    #1, R0              ; 0725
                 04 0009D    RET                     ; 0726

; Routine Size:  158 bytes,    Routine Base:  CODE + 02AB
```

```
;    0727  1 %SBTTL 'CRD$Start_Error'
;    0728  1
;    0729  1 GLOBAL ROUTINE CRD$Start_Error =
;    0730  1
;    0731  1 !++
;    0732  1 ! Functional Description:
;    0733  1 !
;    0734  1 ! There was an error in starting the diagnostic. Print a message telling the operator
;    0735  1 ! this and go to the next diagnostic.
;    0736  1 !
;    0737  1 ! Formal Parameters:
;    0738  1 !
;    0739  1 ! None
;    0740  1 !
;    0741  1 ! Side Effects:
;    0742  1 !
;    0743  1 ! None
;    0744  1 !
;    0745  1 ! Completion Codes:
;    0746  1 !
;    0747  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0748  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0749  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0750  1 !--
```

```
;    0751  2 BEGIN     ! Routine CRD$Start_Error
;    0752  2  MAP
;    0753  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0754  2
;    0755  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed>   = True;   ![19]
;    0756  2      ! We will print a message here.
;    0757  2
;    0758  2  CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete> = False;   ![18]
;    0759  2      ! We did not complete testing on this device.    ![18]
;    0760  2
;    0761  2  !+
;    0762  2  ! Print a message indicating that the diagnostic could not be started for some reason.
;    0763  2  ! Return Repeat_Turing so that the Turing_Machine can do an Advance_State (to the
;    0764  2  ! Advance_Diagnostic state).
;    0765  2  !-
;    0766  2
;    0767  2  $CRD_Print (CRD$GT_Start_Error);
;    0768  2
;    0769  2  CRD$Print_DDB (.CRD$GA_Current_Diagnostic);
;    0770  2
;    0771  3  RETURN (CRD$K_Repeat_Turing)
;    0772  1 END;     ! Routine CRD$Start_Error



          .EXTRN  CRD$GT_START_ERROR

            0000 00000     .ENTRY  CRD$START_ERROR, Save nothing        ; 0729
           50 00000000G  EF  D0 00002     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0755
      14   A0 40    8F  88 00009     BISB2   #64, 20(R0)          ;
      14   A0       20  8A 0000E     BICB2   #32, 20(R0)          ; 0758
       00000000G  EF  9F 00012     PUSHAB  CRD$GT_START_ERROR          ; 0767
            01  DD 00018     PUSHL   #1           ;
            1A  DD 0001A     PUSHL   #26          ;
       00000000G  FF       03  FB 0001C     CALLS   #3, @CRD$PRINT          ;
       00000000G  EF  DD 00023     PUSHL   CRD$GA_CURRENT_DIAGNOSTIC        ; 0769
    FF34   CF       01  FB 00029     CALLS   #1, CRD$PRINT_DDB          ;
           50       02  D0 0002E     MOVL    #2, R0          ; 0771
        04 00031     RET                 ; 0772

; Routine Size:  50 bytes,    Routine Base:  CODE + 0349
```

```
;   0773  1 END     ! End of CRDACT1 module
;   0774  0 ELUDOM




;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          365  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          891  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)




Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
$ASCIC    Macro    Lib03   316     413     716     717     718     719     720     721     722
$CRD_LITERALS   Macro    Lib01   208
$CRD_PRINT   Macro    Lib01   263     309     316     413     553     629     637     660     661     676
             677     681     716     717     718     719     720     721     722     767
$DS_ABORT   KeyWMacr Lib03   318     685
$DS_TYPEDEF   Macro    Lib03   263     309     316     416     553     629     637     660     661     676
             677     681     716     717     718     719     720     721     722     767
$EQULST    Macro    Lib04   208     263     309     316     416     553     629     637     660     661
             676     677     681     716     717     718     719     720     721     722
             767
$ER    Comptime   224
$MODULE    Bind       224
ADDRESS    RoutForm   451   494.    499.    500a.
AUTO$K_EXIT_AUTOSIZER_ERROR Literal    208   267
AUTO$K_EXIT_CONTROL_C  Literal    208
AUTO$K_EXIT_DUMMY_2  Literal    208
AUTO$K_EXIT_DUMMY_3  Literal    208
AUTO$K_EXIT_ERRORS_COMPLETE Literal    208
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal    208   424
AUTO$K_EXIT_INTERNAL_ERROR Literal    208   510
AUTO$K_EXIT_INV_BASE_SYSTEM Literal    208
AUTO$K_EXIT_LAST_REASON  Literal    208   208
AUTO$K_EXIT_NOERRORS_COMPLETE Literal    208   440
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal    208   435
AUTO$K_EXIT_NOERRORS_PREP Literal    208   429
BAD    Register   364   376=    391.
BAD_DEVICE   Register   352   352=    391=    391.    416.    422.
BOOLEAN    Macro    Lib01   399     401
COMPLETE   Register   351   351=    383=    383.    384.    416.    433.
CRD$AL_CRD_DISPATCH_VECTORS External Lib01   476m    485=    485.
CRD$AUTOSIZER_ERROR  GlobRout   227 Lib01e    183f
CRD$DIAGNOSTIC_ERROR  GlobRout   274 Lib01e    186f
CRD$EXIT_CRD   GlobRout   325 Lib01e    189f
CRD$GA_CURRENT_DIAGNOSTIC External Lib01   300m    302a=   303a=   306a=   311a=   313.    489.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Map          539   541a=   544a=   555.
    Map          586   589a.   606a=   607a=   608a=   683.
    Map          753   755a=   758a=   769.
CRD$GA_FIRST_DIAGNOSTIC  External Lib01   348m    357.    410.    490.
CRD$GA_USER_ERROR_TEXT  External Lib01   596.    597a.
CRD$GB_CRD_DEBUG  External Lib01   315.    412.    714.
CRD$GL_CRD_DEBUG_VECTOR  External Lib01   489=    490=    491=    492=    493=    494=    495=    496=    500=    502=
CRD$GL_CURRENT_ACTION  External Lib01   493.
CRD$GL_CURRENT_STATE  External Lib01   492.
CRD$GL_CURRENT_TYPECODE  External Lib01   491.
CRD$GT_AUTOSIZER_ERROR  External Lib01   263a
CRD$GT_CANNOT_LOAD_DIAG  External Lib01   553a
CRD$GT_FAIL   External Lib01   309a
CRD$GT_NEW_LINE   External Lib01   629a    681a
CRD$GT_PREP_ERROR_SPACES External Lib01   637a    660a    661a    676a    677a
CRD$GT_START_ERROR  External Lib01   767a
CRD$INTERNAL_ERROR  GlobRout   451 Lib01e    192f
CRD$K_ACTION_EQL_DO_NOTHING Literal    208
CRD$K_ACTION_RANGE_ERROR Literal    208
CRD$K_ADVANCE_DIAGNOSTIC Literal    208
CRD$K_ADVANCE_GENERAL_COM Literal    208
CRD$K_ADVANCE_STATE  Literal    208
CRD$K_ALLOCATION_ERROR  Literal    208
CRD$K_ASSIGN_PTABLE_PTRS Literal    208
CRD$K_AUTOSIZER_ERROR  Literal    208
CRD$K_AUTOSIZER_SET_FLAGS Literal    208
CRD$K_BAD_ACTION_NUMBER  Literal    208
CRD$K_BAD_TYPECODE_ERROR Literal    208
CRD$K_BASE_SYSTEM_IDC  Literal    208
CRD$K_BASE_SYSTEM_LESI  Literal    208
CRD$K_BASE_SYSTEM_NULL  Literal    208
CRD$K_BASE_SYSTEM_UDA_0  Literal    208
CRD$K_BASE_SYSTEM_UDA_1  Literal    208
CRD$K_BASE_SYSTEM_UDA_2  Literal    208
CRD$K_BASE_SYSTEM_UDA_3  Literal    208
CRD$K_CRD_INITIALIZATION Literal    208
CRD$K_CREATE_HST  Literal    208
CRD$K_DATA_LENGTH_ERROR  Literal    208
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal    208   589     720
CRD$K_DDB_BLINK   Literal    208   718
CRD$K_DDB_FLINK   Literal    208   408     717
CRD$K_DDB_LAST_ENTRY  Literal    208
CRD$K_DDB_MEMORY_SIZE  Literal    208
CRD$K_DDB_MENU_DEVICE_NUMB Literal    208
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal    208   721
CRD$K_DDB_MENU_SUPPORT_SIZE Literal    208
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal    208
CRD$K_DDB_PTABLE_ENTRY  Literal    208   719
CRD$K_DDB_STATUS  Literal    208   302     303     306     311     374     375     376     377     541     544
                606     607     608     722     755     758
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal    208
CRD$K_DEVICE_NAME  Literal    208
CRD$K_DIAGNOSTIC_ERROR  Literal    208
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_DIAGNOSTIC_NAME  Literal    208
CRD$K_DONE_GENERAL_COMS  Literal    208
CRD$K_DO_NOTHING  Literal    208
CRD$K_DUMMY_10   Literal    208
CRD$K_DUMMY_11   Literal    208
CRD$K_DUMMY_12   Literal    208
CRD$K_DUMMY_13   Literal    208
CRD$K_DUMMY_3   Literal    208
CRD$K_DUMMY_4   Literal    208
CRD$K_DUMMY_5   Literal    208
CRD$K_DUMMY_6   Literal    208
CRD$K_DUMMY_7   Literal    208
CRD$K_DUMMY_8   Literal    208
CRD$K_DUMMY_9   Literal    208
CRD$K_EXIT_CRD   Literal    208
CRD$K_HOOK_POINT  Literal    208
CRD$K_HS_INTERNAL_ERROR  Literal    208
CRD$K_IDC_EVDLB   Literal    208
CRD$K_IDC_EVDLC   Literal    208
CRD$K_IDC_EVDLD   Literal    208
CRD$K_IDC_EVRAD_0  Literal    208
CRD$K_IDC_EVRAD_1  Literal    208
CRD$K_IDC_LAST_DIAGNOSTIC Literal    208
CRD$K_INTERNAL_ERROR  Literal    208
CRD$K_LAST_ACTION_ROUTINE Literal    208
CRD$K_LAST_ENTRY  Literal    208
CRD$K_LAST_FUNCTION  Literal    208
CRD$K_LAST_HOOK_POINT  Literal    208
CRD$K_LAST_INTERNAL_ERROR Literal    208
CRD$K_LAST_TYPE   Literal    208
CRD$K_LESI_ENWCA  Literal    208
CRD$K_LESI_EVRMB_0  Literal    208
CRD$K_LESI_EVRMB_1  Literal    208
CRD$K_LESI_LAST_DIAGNOSTIC Literal    208
CRD$K_LEVEL_4_PASSED  Literal    208
CRD$K_LOAD_AUTOSIZER  Literal    208
CRD$K_LOAD_ERROR  Literal    208
CRD$K_LOAD_GENERAL_COM  Literal    208
CRD$K_LOAD_LOAD_COM  Literal    208
CRD$K_LOAD_SELECT_COM  Literal    208
CRD$K_LOAD_SET_EVENT_COM Literal    208
CRD$K_LOAD_SET_FLAGS_COM Literal    208
CRD$K_LOAD_START_COM  Literal    208
CRD$K_LOAD_SUP_FILE  Literal    208
CRD$K_MM_INTERNAL_ERROR  Literal    208
CRD$K_NEW_LINE   Literal    208
CRD$K_PREPARATION_ERROR  Literal    208
CRD$K_PREPARATION_ERROR_TEXT Literal    208   593
CRD$K_REPEAT_TURING  Literal  Lib01   557     771
CRD$K_RUNTIME   Literal    208
CRD$K_SAME_LINE   Literal    208
CRD$K_SET_EVENT_ARGS  Literal    208
CRD$K_SET_FLAGS_ARGS  Literal    208
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------------- ------- ----- ---------------
CRD$K_SET_UP_DIAGNOSTIC  Literal    208
CRD$K_START   Literal    208
CRD$K_START_AUTOSIZER  Literal    208
CRD$K_START_ERROR  Literal    208
CRD$K_START_QUALIFIERS  Literal    208
CRD$K_STAR_LINE   Literal    208
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal    208
CRD$K_STATE_ADVANCE_GENERAL_COM Literal    208
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal    208
CRD$K_STATE_AUTOSIZER_ERROR Literal    208
CRD$K_STATE_AUTOSIZER_RUNNING Literal    208
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal    208
CRD$K_STATE_DIAGNOSTIC_ERROR Literal    208
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal    208
CRD$K_STATE_DONE_DIAGNOSTICS Literal    208
CRD$K_STATE_DONE_GENERAL_COMS Literal    208
CRD$K_STATE_DUMMY_1  Literal    208
CRD$K_STATE_DUMMY_10  Literal    208
CRD$K_STATE_DUMMY_12  Literal    208
CRD$K_STATE_DUMMY_13  Literal    208
CRD$K_STATE_DUMMY_2  Literal    208
CRD$K_STATE_DUMMY_3  Literal    208
CRD$K_STATE_DUMMY_4  Literal    208
CRD$K_STATE_DUMMY_6  Literal    208
CRD$K_STATE_DUMMY_7  Literal    208
CRD$K_STATE_DUMMY_8  Literal    208
CRD$K_STATE_EXIT_CRD  Literal    208
CRD$K_STATE_INTERNAL_ERROR Literal    208
CRD$K_STATE_LAST_STATE  Literal    208
CRD$K_STATE_LEVEL_4_PASSED Literal    208
CRD$K_STATE_LOAD_AUTOSIZER Literal    208
CRD$K_STATE_LOAD_ERROR  Literal    208
CRD$K_STATE_LOAD_GENERAL_COM Literal    208
CRD$K_STATE_LOAD_LOAD_COM Literal    208
CRD$K_STATE_LOAD_SELECT_COM Literal    208
CRD$K_STATE_LOAD_SET_EVENT_COM Literal    208
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal    208
CRD$K_STATE_LOAD_START_COM Literal    208
CRD$K_STATE_PREPARATION_ERROR Literal    208
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal    208
CRD$K_STATE_START  Literal    208
CRD$K_STATE_START_AUTOSIZER Literal    208
CRD$K_STATE_START_ERROR  Literal    208
CRD$K_SUCCESS   Literal  Lib01   270     320     447     511     687     725
CRD$K_TEST_START_TEXT  Literal    208
CRD$K_TQ_INTERNAL_ERROR  Literal    208
CRD$K_TYPECODE   Literal    208
CRD$K_UDA_0_EVDLB  Literal    208
CRD$K_UDA_0_EVDLC  Literal    208
CRD$K_UDA_0_EVDLD  Literal    208
CRD$K_UDA_0_EVRLA_0  Literal    208
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal    208
CRD$K_UDA_1_EVDLB  Literal    208
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_UDA_1_EVDLC  Literal    208
CRD$K_UDA_1_EVDLD  Literal    208
CRD$K_UDA_1_EVRLA_0  Literal    208
CRD$K_UDA_1_EVRLA_1  Literal    208
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal    208
CRD$K_UDA_2_EVDLB  Literal    208
CRD$K_UDA_2_EVDLC  Literal    208
CRD$K_UDA_2_EVDLD  Literal    208
CRD$K_UDA_2_EVRLA_0  Literal    208
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal    208
CRD$K_UDA_3_EVDLB  Literal    208
CRD$K_UDA_3_EVDLC  Literal    208
CRD$K_UDA_3_EVDLD  Literal    208
CRD$K_UDA_3_EVRLA_0  Literal    208
CRD$K_UDA_3_EVRLA_1  Literal    208
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal    208
CRD$LOAD_ERROR   GlobRout   515 Lib01e    195f
CRD$PREPARATION_ERROR  GlobRout   561 Lib01e    198f
CRD$PRINT   External Lib01   263ac   309ac   316ac   416ac   553ac   629ac   637ac   660ac   661ac   676ac
             677ac   681ac   716ac   717ac   718ac   719ac   720ac   721ac   722ac   767ac
CRD$PRINT_DDB   GlobRout   691 Lib01e    201f    313c    372c    555c    683c    769c
CRD$START_ERROR   GlobRout   729 Lib01e    204f
CRD$STOP   ExtRout  Lib01   267c    424c    429c    435c    440c    510c
DDB$V_STATUS_DEVICE_BAD  Macro    Lib01   303     376
DDB$V_STATUS_DEV_TEST_COMPLETE Macro    Lib01   306     374     544     608     758
DDB$V_STATUS_DIAGNOSTIC_ERROR Macro    Lib01   302
DDB$V_STATUS_ESSENTIAL_DEVICE Macro    Lib01   375
DDB$V_STATUS_MESSAGE_PRINTED Macro    Lib01   311     541     607     755
DDB$V_STATUS_PREPARATION_ERROR Macro    Lib01   377     606
DDB_PTR    Register   355   357=    372.    374a.   375a.   376a.   377a.   408=    408a.   410.
    RoutForm   691   712m    716.    717a.   718a.   719a.   720a.   721a.   722a.
DEVICE_DATA_BLOCK_STRUCTURE Structur Lib01   300     348     355     539     586     712     753
DIAG_VECTOR_PTR   Bind        589   593.
DIFFERENCE   Local       642   644=
DS$ABORT   ExtRout    318   318c    685e    685c
DS$K_PRINTB   Literal    263
    Literal    309
    Literal    316
    Literal    416
    Literal    553
    Literal    629
    Literal    637
    Literal    660
    Literal    661
    Literal    676
    Literal    677
    Literal    681
    Literal    716
    Literal    717
    Literal    718
    Literal    719
    Literal    720
    Literal    721
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     722
    Literal     767
DS$K_PRINTF   Literal     263   263
    Literal     309   309
    Literal     316   316
    Literal     416   416
    Literal     553   553
    Literal     629   629
    Literal     637   637
    Literal     660   660
    Literal     661   661
    Literal     676   676
    Literal     677   677
    Literal     681   681
    Literal     716   716
    Literal     717   717
    Literal     718   718
    Literal     719   719
    Literal     720   720
    Literal     721   721
    Literal     722   722
    Literal     767   767
DS$K_PRINTI   Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_PRINTX   Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ABORT_PROGRAM  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ABORT_TEST  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_COMMAND_ERR  Literal     263
    Literal     309
    Literal     316
    Literal     416
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_COMMAND_OUT  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_CRD_AUTOTEST  Literal     263   263
    Literal     309   309
    Literal     316   316
    Literal     416   416
    Literal     553   553
    Literal     629   629
    Literal     637   637
    Literal     660   660
    Literal     661   661
    Literal     676   676
    Literal     677   677
    Literal     681   681
    Literal     716   716
    Literal     717   717
    Literal     718   718
    Literal     719   719
    Literal     720   720
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     721    721
    Literal     722    722
    Literal     767    767
DS$K_TYPE_DS_PROMPT  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_DS_START  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ERRDEV  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ERRHARD  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ERROR_BODY  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ERROR_END  Literal     263
    Literal     309
    Literal     316
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ERRPREP  Literal    263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ERRSOFT  Literal    263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ERRSUP  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ERRSYS  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_ERR_HALT  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
```

Symbol	Type	Defined	Referenced ...
	Literal		676
	Literal		677
	Literal		681
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		720
	Literal		721
	Literal		722
	Literal		767
DS$K_TYPE_EXCEPTION	Literal		263
	Literal		309
	Literal		316
	Literal		416
	Literal		553
	Literal		629
	Literal		637
	Literal		660
	Literal		661
	Literal		676
	Literal		677
	Literal		681
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		720
	Literal		721
	Literal		722
	Literal		767
DS$K_TYPE_EXCEPTION_HEAD	Literal		263
	Literal		309
	Literal		316
	Literal		416
	Literal		553
	Literal		629
	Literal		637
	Literal		660
	Literal		661
	Literal		676
	Literal		677
	Literal		681
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		720
	Literal		721
	Literal		722
	Literal		767
DS$K_TYPE_FIRST_PASS	Literal		263
	Literal		309

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_GENERAL  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_GENERAL_ERROR  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_NO_TESTS  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_PARAM_ERROR  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_PROGRAM_END  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- --- - ---------------
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_PROGRAM_INFO  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_PROGRAM_START  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_QIO_INVADP  Literal     263
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_QIO_NODRIVER  Literal    263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_QIO_WRONGVER  Literal    263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_SCRIPT_ECHO  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_SCRIPT_PNF  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_SCRIPT_PROMPT  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_SCRIPT_SKIP  Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_SEQUENCE_ERROR Literal     263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DS$K_TYPE_START_ERR  Literal    263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_START_LIST  Literal    263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_SUMMARY  Literal    263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
DS$K_TYPE_USER_PROMPT  Literal    263
    Literal     309
    Literal     316
    Literal     416
    Literal     553
    Literal     629
    Literal     637
    Literal     660
    Literal     661
    Literal     676
    Literal     677
    Literal     681
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     720
    Literal     721
    Literal     722
    Literal     767
ERROR_MESSAGE_ADDRESS  RoutForm   274
ERROR_MESSAGE_LENGTH  RoutForm   274
ESSENTIAL_DEVICE  Register   365   375=    384.    399.
FALSE    Literal  Lib01   306     352     353     544     600     608     758
FINISHED   Register   366   374=    383.    384.
GET1ST_    Macro    Lib04   208     263     309     316     416     553     629     637     660     661
              676     677     681     716     717     718     719     720     721     722
              767
GET2ND_    Unbound          208     263     309     316     416     553     629     637     660     661
              676     677     681     716     717     718     719     720     721     722
              767
HARDWARE_SUPPORT_VECTOR  Structur Lib01   590
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal    208
HS$K_ACTION_ADVANCE_GENERAL_COM Literal    208
HS$K_ACTION_ADVANCE_STATE Literal    208
HS$K_ACTION_CHANGE_TABLE Literal    208
HS$K_ACTION_DIAGNOSTIC_ERROR Literal    208
HS$K_ACTION_DONE_GENERAL_COMS Literal    208
HS$K_ACTION_DO_NOTHING  Literal    208
HS$K_ACTION_DUMMY_3  Literal    208
HS$K_ACTION_DUMMY_4  Literal    208
HS$K_ACTION_DUMMY_5  Literal    208
HS$K_ACTION_DUMMY_6  Literal    208
HS$K_ACTION_INIT_HS  Literal    208
HS$K_ACTION_INTERNAL_ERROR Literal    208
HS$K_ACTION_LAST_ACTION  Literal    208
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_ACTION_LOAD_ERROR  Literal    208
HS$K_ACTION_LOAD_GENERAL_COM Literal    208
HS$K_ACTION_LOAD_LOAD_COM Literal    208
HS$K_ACTION_LOAD_SELECT_COM Literal    208
HS$K_ACTION_LOAD_SET_EVENT_COM Literal    208
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal    208
HS$K_ACTION_LOAD_START_COM Literal    208
HS$K_ACTION_PREPARATION_ERROR Literal    208
HS$K_ACTION_START_ERROR  Literal    208
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal    208
HS$K_STATE_ADVANCE_GENERAL_COM Literal    208
HS$K_STATE_CHANGE_TABLE  Literal    208
HS$K_STATE_DIAGNOSTIC_ERROR Literal    208
HS$K_STATE_DIAGNOSTIC_RUNNING Literal    208
HS$K_STATE_DONE_GENERAL_COMS Literal    208
HS$K_STATE_DUMMY_1  Literal    208
HS$K_STATE_DUMMY_2  Literal    208
HS$K_STATE_DUMMY_3  Literal    208
HS$K_STATE_DUMMY_4  Literal    208
HS$K_STATE_DUMMY_6  Literal    208
HS$K_STATE_INIT_HS  Literal    208
HS$K_STATE_INTERNAL_ERROR Literal    208
HS$K_STATE_LAST_STATE  Literal    208
HS$K_STATE_LOAD_ERROR  Literal    208
HS$K_STATE_LOAD_GENERAL_COM Literal    208
HS$K_STATE_LOAD_LOAD_COM Literal    208
HS$K_STATE_LOAD_SELECT_COM Literal    208
HS$K_STATE_LOAD_SET_EVENT_COM Literal    208
HS$K_STATE_LOAD_SET_FLAGS_COM Literal    208
HS$K_STATE_LOAD_START_COM Literal    208
HS$K_STATE_PREPARATION_ERROR Literal    208
HS$K_STATE_START_ERROR  Literal    208
INCORRECT_PREP   Register   353   353=    397=    397.    416.    427.
LEFT    Local      653   658=
    Local      670   674=
LENGTH    RoutForm   451   495.
MEN$K_HS_STATE_TABLE  Literal    208
MEN$K_MM_STATE_TABLE  Literal    208
MEN$K_TQ_STATE_TABLE  Literal    208
MENU$K_EXIT_AUTOSIZER_ERROR Literal    208
MENU$K_EXIT_INTERNAL_ERROR Literal    208
MENU$K_EXIT_LAST_REASON  Literal    208
MENU$K_EXIT_OPERATOR_ABORT Literal    208
MENU$K_EXIT_OPERATOR_STOP Literal    208
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal    208
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal    208
MM$K_ACTION_ADVANCE_STATE Literal    208
MM$K_ACTION_AUTOSIZER_ERROR Literal    208
MM$K_ACTION_BUILD_DDBS  Literal    208
MM$K_ACTION_BUILD_HSTS  Literal    208
MM$K_ACTION_CHANGE_TABLE Literal    208
MM$K_ACTION_DONE_INITS  Literal    208
MM$K_ACTION_DO_NOTHING  Literal    208
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_ACTION_DUMMY_3  Literal    208
MM$K_ACTION_DUMMY_4  Literal    208
MM$K_ACTION_DUMMY_5  Literal    208
MM$K_ACTION_DUMMY_6  Literal    208
MM$K_ACTION_DUMMY_7  Literal    208
MM$K_ACTION_DUMMY_8  Literal    208
MM$K_ACTION_INTERNAL_ERROR Literal    208
MM$K_ACTION_LAST_ACTION  Literal    208
MM$K_ACTION_LOAD_AUTOSIZER Literal    208
MM$K_ACTION_MENU_PROMPT  Literal    208
MM$K_ACTION_PRINT_MENU  Literal    208
MM$K_ACTION_PRINT_MENU_HEADING Literal    208
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal    208
MM$K_ACTION_START_AUTOSIZER Literal    208
MM$K_STATE_AUTOSIZER_ERROR Literal    208
MM$K_STATE_AUTOSIZER_RUNNING Literal    208
MM$K_STATE_BUILD_DDBS  Literal    208
MM$K_STATE_BUILD_HSTS  Literal    208
MM$K_STATE_CHANGE_TABLE  Literal    208
MM$K_STATE_DONE_INITS  Literal    208
MM$K_STATE_DUMMY_1  Literal    208
MM$K_STATE_DUMMY_2  Literal    208
MM$K_STATE_DUMMY_3  Literal    208
MM$K_STATE_DUMMY_5  Literal    208
MM$K_STATE_DUMMY_7  Literal    208
MM$K_STATE_DUMMY_8  Literal    208
MM$K_STATE_INTERNAL_ERROR Literal    208
MM$K_STATE_LAST_STATE  Literal    208
MM$K_STATE_LOAD_AUTOSIZER Literal    208
MM$K_STATE_MENU_PROMPT  Literal    208
MM$K_STATE_PRINT_MENU  Literal    208
MM$K_STATE_PRINT_MENU_HEADING Literal    208
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal    208
MM$K_STATE_START  Literal    208
MM$K_STATE_START_AUTOSIZER Literal    208
MODNAM    Macro    Lib02   224
NO_USER_TEXT   Local      600   600=    623=    625.    627=    631.
NUL2ND_    Macro    Lib04   208     263     309     316     416     553     629     637     660     661
              676     677     681     716     717     718     719     720     721     722
              767
PREP_ERROR   Register   363   377=    401.
PREP_ERROR_TEXT   Bind       593   637a    644.    646.    661a    677a
RIGHT    Local      653   657=
    Local      670   675=
TQ$K_ACTION_ADVANCE_STATE Literal    208
TQ$K_ACTION_CHANGE_TABLE Literal    208
TQ$K_ACTION_DONE_TEST_QUEUE Literal    208
TQ$K_ACTION_DO_NOTHING  Literal    208
TQ$K_ACTION_DUMMY_3  Literal    208
TQ$K_ACTION_DUMMY_4  Literal    208
TQ$K_ACTION_DUMMY_5  Literal    208
TQ$K_ACTION_GET_DDB  Literal    208
TQ$K_ACTION_INIT_TQ  Literal    208
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
TQ$K_ACTION_INTERNAL_ERROR Literal    208
TQ$K_ACTION_LAST_ACTION  Literal    208
TQ$K_STATE_CHANGE_TABLE  Literal    208
TQ$K_STATE_DONE_TEST_QUEUE Literal    208
TQ$K_STATE_DUMMY_1  Literal    208
TQ$K_STATE_DUMMY_2  Literal    208
TQ$K_STATE_DUMMY_3  Literal    208
TQ$K_STATE_DUMMY_4  Literal    208
TQ$K_STATE_DUMMY_5  Literal    208
TQ$K_STATE_GET_DDB  Literal    208
TQ$K_STATE_INIT_TQ  Literal    208
TQ$K_STATE_INTERNAL_ERROR Literal    208
TQ$K_STATE_LAST_STATE  Literal    208
TRUE    Literal  Lib01   302     303     311     351     541     606     607     623     627     755
TYPECODE   RoutForm   274
    RoutForm   451   496.
USER_ERROR_TEXT   Bind       597   626.    644.    646.    660a    676a
USER_ERROR_TEXT_ADDRESS  Bind       596   622.


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDACT1.B32;1
    2  Module CRDACT1
  170  Library #1 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
  173  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
  176  Library #3 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
  179  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  774  Eludom CRDACT1


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     a Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
```

```
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;          486        40         8        62          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;          671         1         0        46          00:00.2
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;          923         3         0        94          00:00.2
;  SYS$COMMON:[SYSLIB]LIB.L32;2       18619          3          0      1000          00:04.2




;     COMMAND QUALIFIERS

; BLISS CRDACT1.B32/LIST=CRDACT1.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDACT1.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  891 code + 365 data bytes
; Run Time:    00:58.8
; Elapsed Time:    01:32.3
; Lines/CPU Min:     790
; Lexemes/CPU-Min: 72164
; Memory Used: 290 pages
; Compilation Complete
```

```
;    0001  0 %TITLE '*** CRDACT2 Module'
;    0002  0 MODULE CRDACT2 (    ! Routines to perform actions on DS typecodes
;    0003  0   IDENT = '01-17'
;    0004  0   ) =
;    0005  0 !++
;    0006  0 ! COPYRIGHT (c) 1980, 1981
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! This module contains the routines necessary to perform actions based upon
;    0032  0 ! a DS typecode and a current "state". A state table is used to determine what
;    0033  0 ! action routine should be undertaken.
;    0034  0 !
;    0035  0 ! Author:
;    0036  0 !
;    0037  0 ! Jack Stansbury
;    0038  0 !
;    0039  0 ! Creation Date:
;    0040  0 !
;    0041  0 ! 23-April-1982
;    0042  0 !
;    0043  0 ! Version:
;    0044  0 !
;    0045  0 ! 6.9
;    0046  0 !
;    0047  0 ! Modified By:
;    0048  0 !
;    0049  0 ! 01 Jack Stansbury, Version 1.0, June 21, 1982 (Summer!!!)
;    0050  0 !  Changed CRD$F_DDB to CRD$V_DDB. Also added functionality of the
;    0051  0 !  DDB$V_Status_End_Testing bit. This bit indicates that everything
;    0052  0 !  has been done on that device.
;    0053  0 !
;    0054  0 ! 02 Jack Stansbury, Version 1.0, June 22, 1982
;    0055  0 !  Took out the Error_Count variables. These are not needed for this version.
;    0056  0 !  Also added some functionality to the Internal_Error routine to allow it
;    0057  0 !  to be used as a general purpose internal error routine (i.e., callable from
;    0058  0 !  anywhere in CRD-AutoTest).
;    0059  0 !
;    0060  0 ! 03 Jack Stansbury, Version 1.0, July 2, 1982
;    0061  0 !  Made changes to the Internal_Error routine so that it will not overwrite
;    0062  0 !  the dispatch vectors before the vectors are no longer needed (i.e., CRD$Stop
;    0063  0 !  needs at least the CRD$Print address in the vectors to print things). Use
;    0064  0 !  a new global vector to store the things in that will later be transferred
;    0065  0 !  to the dispatch area by CRD$Stop.
;    0066  0 !
;    0067  0 ! 04 Jack Stansbury, Version 1.0, July 7, 1982
;    0068  0 !  Added code in the CRD$Preparation_Error routine that will do special case
;    0069  0 !  code for printing the preparation error message. This was necessary because
;    0070  0 !  of the way the message was printed when the user_error text was one character
;    0071  0 !  less in length than the prep_error text. This caused an asterisk to be left
;    0072  0 !  out. I'm not sure if this will fix the problem, but it may help.
;    0073  0 !
;    0074  0 ! 05 Jack Stansbury, Version 1.0, July 9, 1982
;    0075  0 !  Decrement the positive CRD version number when there is an internal error.
;    0076  0 !  This is contrary to the documentation in the DSV$Exit routine because the
;    0077  0 !  code in the routine is wrong (i.e., it does an INCL when it should have done
;    0078  0 !  a DECL). This will have to be changed in the DSV$Exit routine, and in the
```

```
;    0079  0 !  CRD$Internal_Error routine later - after CRD-AutoTest goes out.
;    0080  0 !
;    0081  0 ! 06 Jack Stansbury, Version 1.0, July 9, 1982
;    0082  0 !  Make changes to the CRD$Diagnostic_Error routine. It was causing a problem because
;    0083  0 !  I had made a change in the state_table (making a Script_Prompt typecode activate
;    0084  0 !  the Diagnostic_Error routine). The SELECTONE statement in the Internal_Error routine
;    0085  0 !  did not know this new typecode, and it reported an internal error. So, I took out
;    0086  0 !  the SELECTONE statement because it is really not needed. It was in there to keep
;    0087  0 !  track of the number of different types of errors that occurred. I don't do that
;    0088  0 !  anymore.
;    0089  0 !
;    0090  0 ! 07 Jack Stansbury, Version 1.0, July 9, 1982
;    0091  0 !  Make some changes in the Start_Error and Load_Error routines that will make the ending
;    0092  0 !  message come out right. When one start error occurs, the ending message should be the
;    0093  0 !  check device preparation message. To get this to come out, I set the Status_Essential_Dev
;    0094  0 !  bit equal to True. That way, the Exit_CRD routine will think that an essential device
;    0095  0 !  wasn't tested, and print the right message.
;    0096  0 !
;    0097  0 ! 08 Jack Stansbury, Version 1.0, July 9, 1982
;    0098  0 !  Changed the cose in the Preparation_Error routine to handle the special case of the
;    0099  0 !  '!#*' FAO construct when the parameter for the '#' is 0. The DS's FAO routine do not
;    0100  0 !  handle this correctly. They eat a character out of the FAO control string. So, I have
;    0101  0 !  to check for this when I print the preparation error messaes, and compensate for it.
;    0102  0 !
;    0103  0 ! 09 Jack Stansbury, Version 1.1, July 16, 1982
;    0104  0 !  Added functionality for CRD Menu Test.
;    0105  0 !
;    0106  0 ! 10 Jack Stansbury, Version 1.1, July 23, 1982
;    0107  0 !  Fix what is documented in [05] above.
;    0108  0 !
;    0109  0 ! 11 Jack Stansbury, Version 6.9, July 25, 1982
;    0110  0 !  Changed the Preparation error messages back to what they should have been (I fixed the
;    0111  0 !  FAO routines).
;    0112  0 !
;    0113  0 ! 12 Jack Stansbury, Version 1.1, July 25, 1982
;    0114  0 !  Split the CRDACTION module into two different modules: CRDACT1 and CRDACT2. They both
;    0115  0 !  retain the modification history of CRDACTION, but these modification historys become
;    0116  0 !  particular to the module after this point. This module contains the action routines
;    0117  0 !  that don't concern errors or exiting. This does not contain any global variables either.
;    0118  0 !
;    0119  0 ! 13 Jack Stansbury, Version 1.1, August 20, 1982
;    0120  0 !  Changed the field names for the DDBs, and made changes in the AutoSizer handling.
;    0121  0 !
;    0122  0 ! 14 Jack Stansbury, Version 1.1, August 31, 1982
;    0123  0 !  Added the Complete status bit to the DDB status handling in Advance_Diagnostic.
;    0124  0 !
;    0125  0 ! 15 Jack Stansbury, 15-September-1982, Version 1.1
;    0126  0 !  Added calls to the CRD$Print_DS_Command routine which will print the DS command currently
;    0127  0 !  in the CLI command buffer. This is done only when the DSA$V_CRD_Trace flag is set.
;    0128  0 !
;    0129  0 ! 16 Jack Stansbury, 30-September-1982, Version 1.1
;    0130  0 !  Changed the state and action routine numbers all around - made them like the Menu states
;    0131  0 !  and action numbers. Added two more action routines for Auto Test.
;    0132  0 !
;    0133  0 ! 17 John Ciukaj 7-Apr-1983 Version 1.2
```

```
;    0134  0 !  - Enhance CRD$Assign_PTable_Ptrs to:
;    0135  0 !    1. specify base system type
;    0136  0 !    2. specify hardware support table to be used
;    0137  0 !    3. Build DDB's
;    0138  0 !     It is assumed that this routine is invoked following
;    0139  0 !     completion of Autosizer.
;    0140  0 !
;    0141  0 !--
```

```
;    0142  0 %SBTTL 'Libraries'
;    0143  1 BEGIN     ! Begin the CRDACT2 module
;    0144  1
;    0145  1 LIBRARY
;    0146  1   '$CRD';
;    0147  1
;    0148  1 LIBRARY
;    0149  1   '$DS';
;    0150  1
;    0151  1 LIBRARY
;    0152  1   '$Diag';
;    0153  1
;    0154  1 LIBRARY
;    0155  1   'Sys$Library:Lib';
```

```
;    0156  1 %SBTTL 'Table of Contents'
;    0157  1
;    0158  1 FORWARD ROUTINE    ! In alphabetical order
;    0159  1
;    0160  1  CRD$Advance_Diagnostic  : ADDRESSING_MODE (LONG_RELATIVE),
;    0161  1      ! Advance to the next diagnostic
;    0162  1
;    0163  1  CRD$Advance_General_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0164  1      ! Go to the next general command
;    0165  1
;    0166  1  CRD$Assign_PTable_Ptrs  : ADDRESSING_MODE (LONG_RELATIVE),
;    0167  1      ! Insert the PTable pointers into the DDBs
;    0168  1
;    0169  1  CRD$AutoSizer_Set_Flags  : ADDRESSING_MODE (LONG_RELATIVE),
;    0170  1      ! Load the SET FLAGS command for the AutoSizer.
;    0171  1  CRD$Determine_Base_System : ADDRESSING_MODE (LONG_RELATIVE),
;    0172  1     ![17]
;    0173  1
;    0174  1  CRD$Done_General_Coms  : ADDRESSING_MODE (LONG_RELATIVE),
;    0175  1      ! All done executing the set of general commands
;    0176  1
;    0177  1  CRD$Level_4_Passed  : ADDRESSING_MODE (LONG_RELATIVE),     ![16]
;    0178  1      ! Print 'PASSED' for the level 4 diagnostic that runs previous  ![16]
;    0179  1      ! to CRD Auto Test - Macro      ![16]
;    0180  1
;    0181  1  CRD$Load_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE),
;    0182  1      ! Load the AutoSizer
;    0183  1
;    0184  1  CRD$Load_General_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0185  1      ! Load a general command into the CLI data buffer
```

```
;    0186  1  CRD$Load_Load_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0187  1      ! Load a LOAD command into the CLI buffer.
;    0188  1
;    0189  1  CRD$Load_Select_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0190  1      ! Load a SELECT command into the CLI buffer.
;    0191  1
;    0192  1  CRD$Load_Set_Event_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0193  1      ! Load a SET EVENT command into the CLI buffer.
;    0194  1
;    0195  1  CRD$Load_Set_Flags_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0196  1      ! Load a SET FLAGS command into the CLI buffer.
;    0197  1
;    0198  1  CRD$Load_Start_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0199  1      ! Load a START command into the CLI buffer.
;    0200  1
;    0201  1  CRD$Set_Up_Diagnostic  : ADDRESSING_MODE (LONG_RELATIVE),     ![16]
;    0202  1      ! Print the testing started message, and check to see of the  ![16]
;    0203  1      ! device is attached. If not attached, print message and go to   ![16]
;    0204  1      ! advance_diagnostic.       ![16]
;    0205  1
;    0206  1  CRD$Start_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE);
;    0207  1      ! Start the AutoSizer executing
```

```
;    0208  1 %SBTTL 'Included Macros and Literals'
;    0209  1
;    0210  1  $DS_TypeDef;    ! Define the DS typecodes
;    0211  1  $CRD_Literals;   ! Define some literals for CRD
```

```
;    0212  1 %SBTTL 'Psect Definitions'
;    0213  1
;    0214  1 PSECT
;    0215  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0216  1
;    0217  1 PSECT
;    0218  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0219  1
;    0220  1 PSECT
;    0221  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0222  1
;    0223  1 PSECT
;    0224  1     GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0225  1 %SBTTL 'Module-Global Static Storage'
;    0226  1
;    0227  1  ModNam ('CRDACT2');  ! Module name for ErrSup macro
```

```
;    0228  1 %SBTTL 'CRD$Advance_Diagnostic'
;    0229  1
;    0230  1 GLOBAL ROUTINE CRD$Advance_Diagnostic =          ![16]
;    0231  1                 ! |
;    0232  1 !++             ! V
;    0233  1 ! Functional Description:
;    0234  1 !
;    0235  1 ! Advance the diagnostic pointer. If it is at the end of the list
;    0236  1 ! of diagnostics, go to done_diagnostics. If not, return repeat turing.
;    0237  1 !
;    0238  1 ! Formal Parameters:
;    0239  1 !
;    0240  1 ! None
;    0241  1 !
;    0242  1 ! Side Effects:
;    0243  1 !
;    0244  1 ! None
;    0245  1 !
;    0246  1 ! Completion Codes:
;    0247  1 !
;    0248  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0249  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0250  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0251  1 !--
```

```
;    0252  2 BEGIN     ! Routine CRD$Advance_Diagnostic
;    0253  2  MAP
;    0254  2   CRD$GA_First_Diagnostic   : REF Device_Data_Block_Structure,
;    0255  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0256  2
;    0257  2  !+
;    0258  2  ! If a message was not printed for the test just completed, print passed.
;    0259  2  !-
;    0260  2
;    0261  2  IF (NOT .CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed>) THEN
;    0262  2   $CRD_Print (CRD$GT_Pass);
;    0263  2
;    0264  2  IF (.CRD$GB_CRD_Debug) THEN
;    0265  2   $CRD_Print ($ASCIC ('Advance_Diagnostic: before advancing!/'));
;    0266  2
;    0267  2  CRD$Print_DDB (.CRD$GA_Current_Diagnostic);
;    0268  2
;    0269  2  !+
;    0270  2  ! We are all done doing things for the current diagnostic. Advance to the next queue entry
;    0271  2  ! in the DDB queue.
;    0272  2  !-
;    0273  2
;    0274  2  CRD$GA_Current_Diagnostic = .CRD$GA_Current_Diagnostic [CRD$K_DDB_FLink];
;    0275  2
;    0276  2  IF (.CRD$GB_CRD_Debug) THEN
;    0277  2   $CRD_Print ($ASCIC ('Advance_Diagnostic: after advancing!/'));
;    0278  2
;    0279  2  CRD$Print_DDB (.CRD$GA_Current_Diagnostic);
;    0280  2
     0281  2  !+
;    0282  2  ! If we are back to the first one in the queue, then change the state number to
;    0283  2  ! Done_Diagnostics and Repeat_Turing to finish off CRD.
;    0284  2  !-
;    0285  2
;    0286  2  IF (.CRD$GA_Current_Diagnostic EQL .CRD$GA_First_Diagnostic) THEN
;    0287  2   CRD$GL_Current_State = CRD$K_State_Done_Diagnostics;      ! ^
;    0288  2                      ! |
;    0289  3   RETURN (CRD$K_Repeat_Turing)          ![16]
;    0290  1 END;     ! Routine CRD$Advance_Diagnostic


          .TITLE  CRDACT2 *** CRDACT2 Module
          .IDENT  \01-17\

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

       32  54  43  41  44  52  43  07  00000 P.AAA:  .ASCII  <7>\CRDACT2\          ;
6F  6E  67  61  69  44  5F  65  63  6E  61  76  64  41  26  00008 P.AAB:  .ASCII  \&Advance_Diagnostic: before advancing!/\   ;
64  61  20  65  72  6F  66  65  62  20  3A  63  69  74  73  00017            ;
   2F  21  67  6E  69  63  6E  61  76  00026              ;
6F  6E  67  61  69  44  5F  65  63  6E  61  76  64  41  25  0002F P.AAC:  .ASCII  \%Advance_Diagnostic: after advancing!/\    ;
76  64  61  20  72  65  74  66  61  20  3A  63  69  74  73  0003E            ;
       2F  21  67  6E  69  63  6E  61  0004D              ;

          $MODULE-       P.AAA
```

```
            .EXTRN  CRD$ADVANCE_DIAGNOSTIC
            .EXTRN  CRD$ADVANCE_GENERAL_COM
            .EXTRN  CRD$ASSIGN_PTABLE_PTRS
            .EXTRN  CRD$AUTOSIZER_SET_FLAGS
            .EXTRN  CRD$DONE_GENERAL_COMS
            .EXTRN  CRD$LEVEL_4_PASSED
            .EXTRN  CRD$LOAD_AUTOSIZER
            .EXTRN  CRD$LOAD_GENERAL_COM
            .EXTRN  CRD$LOAD_LOAD_COM
            .EXTRN  CRD$LOAD_SELECT_COM
            .EXTRN  CRD$LOAD_SET_EVENT_COM
            .EXTRN  CRD$LOAD_SET_FLAGS_COM
            .EXTRN  CRD$LOAD_START_COM
            .EXTRN  CRD$SET_UP_DIAGNOSTIC
            .EXTRN  CRD$START_AUTOSIZER
            .EXTRN  CRD$GA_FIRST_DIAGNOSTIC
            .EXTRN  CRD$GA_CURRENT_DIAGNOSTIC
            .EXTRN  CRD$GT_PASS, CRD$PRINT
            .EXTRN  CRD$GB_CRD_DEBUG
            .EXTRN  CRD$PRINT_DDB, CRD$GL_CURRENT_STATE

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0000 00000     .ENTRY  CRD$ADVANCE_DIAGNOSTIC, Save nothing        ; 0230
            50 00000000G   EF  D0 00002     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0261
            11   14   A0      06  E0 00009     BBS     #6, 20(R0), 1$          ;
         00000000G   EF   9F 0000E     PUSHAB  CRD$GT_PASS             ; 0262
            01   DD 00014     PUSHL   #1              ;
            1A   DD 00016     PUSHL   #26             ;
         00000000G   FF       03  FB 00018     CALLS   #3, @CRD$PRINT          ;
            11 00000000G   EF  E9 0001F 1$:    BLBC    CRD$GB_CRD_DEBUG, 2$         ; 0264
         00000000'   EF   9F 00026     PUSHAB  P.AAB               ; 0265
            01   DD 0002C     PUSHL   #1              ;
            1A   DD 0002E     PUSHL   #26             ;
         00000000G   FF       03  FB 00030     CALLS   #3, @CRD$PRINT          ;
         00000000G   EF   DD 00037 2$:    PUSHL   CRD$GA_CURRENT_DIAGNOSTIC       ; 0267
         00000000G   EF       01  FB 0003D     CALLS   #1, CRD$PRINT_DDB           ;
         00000000G   EF 00000000G   FF   D0 00044     MOVL    @CRD$GA_CURRENT_DIAGNOSTIC, -        ; 0274
              CRD$GA_CURRENT_DIAGNOSTIC          ;
            11 00000000G   EF   E9 0004F     BLBC    CRD$GB_CRD_DEBUG, 3$         ; 0276
         00000000'   EF   9F 00056     PUSHAB  P.AAC               ; 0277
            01   DD 0005C     PUSHL   #1              ;
            1A   DD 0005E     PUSHL   #26             ;
         00000000G   FF       03  FB 00060     CALLS   #3, @CRD$PRINT          ;
         00000000G   EF   DD 00067 3$:    PUSHL   CRD$GA_CURRENT_DIAGNOSTIC       ; 0279
         00000000G   EF       01  FB 0006D     CALLS   #1, CRD$PRINT_DDB           ;
         00000000G   EF 00000000G   EF   D1 00074     CMPL    CRD$GA_CURRENT_DIAGNOSTIC, -        ; 0286
              CRD$GA_FIRST_DIAGNOSTIC            ;
            07   12 0007F     BNEQ    4$              ;
         00000000G   EF       1A  D0 00081     MOVL    #26, CRD$GL_CURRENT_STATE       ; 0287
            50       02  D0 00088 4$:    MOVL    #2, R0              ; 0289
             04 0008B     RET                 ; 0290

; Routine Size:  140 bytes,    Routine Base:  CODE + 0000
```

```
;    0291  1 %SBTTL 'CRD$Advance_General_Com'
;    0292  1
;    0293  1 GLOBAL ROUTINE CRD$Advance_General_Com =
;    0294  1
;    0295  1 !++
;    0296  1 ! Functional Description:
;    0297  1 !
;    0298  1 ! Advance the general command pointer. If it is at the end of the list
;    0299  1 ! of general commands, return failure. Otherwise, return success.
;    0300  1 !
;    0301  1 ! Formal Parameters:
;    0302  1 !
;    0303  1 ! None
;    0304  1 !
;    0305  1 ! Side Effects:
;    0306  1 !
;    0307  1 ! None
;    0308  1 !
;    0309  1 ! Completion Codes:
;    0310  1 !
;    0311  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0312  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0313  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0314  1 !--
```

```
;    0315  2 BEGIN     ! Routine CRD$Advance_General_Com
;    0316  2
;    0317  2  IF (.CRD$GB_CRD_Debug) THEN
; P  0318  2   $CRD_Print ($ASCIC ('**** Advance_General_Com: Current = !XL!/'),
;    0319  2    .CRD$GL_Current_General_Com);
;    0320  2
;    0321  2  IF (.CRD$GL_Current_General_Com EQL (CRD$K_Numb_General_Commands - 1)) THEN
;    0322  3   BEGIN
;    0323  3   !+
;    0324  3   ! We are all done the list of general commands. Set the current state to a special
;    0325  3   ! number that indicates we are done executing these general commands.
;    0326  3   !-
;    0327  3   CRD$GL_Current_State = CRD$K_State_Done_General_Coms;
;    0328  3   END
;    0329  2  ELSE
;    0330  3   BEGIN
;    0331  3   !+
;    0332  3   ! There are more general commands to execute. Get the next one.
;    0333  3   !-
;    0334  3   CRD$GL_Current_General_Com = .CRD$GL_Current_General_Com + 1;
;    0335  2   END;
;    0336  2
;    0337  3  RETURN (CRD$K_Repeat_Turing)
;    0338  1 END;     ! Routine CRD$Advance_General_Com



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

47  5F  65  63  6E  61  76  64  41  20  2A  2A  2A  2A  29  00055 P.AAD:  .ASCII  \)**** Advance_General_Com: Current = !XL\  ;
72  75  43  20  3A  6D  6F  43  5F  6C  61  72  65  6E  65  00064            ;
      4C  58  21  20  3D  20  74  6E  65  72  00073            ;
          2F  21  0007D    .ASCII  \!/\           ;

            .EXTRN  CRD$GL_CURRENT_GENERAL_COM

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0000 00000     .ENTRY  CRD$ADVANCE_GENERAL_COM, Save nothing       ; 0293
            17 00000000G  EF  E9 00002     BLBC    CRD$GB_CRD_DEBUG, 1$         ; 0317
       00000000G  EF  DD 00009     PUSHL   CRD$GL_CURRENT_GENERAL_COM        ; 0319
       00000000'  EF  9F 0000F     PUSHAB  P.AAD           ;
            01  DD 00015     PUSHL   #1            ;
            1A  DD 00017     PUSHL   #26           ;
       00000000G  FF      04  FB 00019     CALLS   #4, @CRD$PRINT          ;
            02 00000000G  EF  D1 00020 1$:   CMPL    CRD$GL_CURRENT_GENERAL_COM, #2       ; 0321
            09  12 00027     BNEQ    2$            ;
       00000000G  EF      10  D0 00029     MOVL    #16, CRD$GL_CURRENT_STATE        ; 0327
            06  11 00030     BRB     3$            ; 0321
       00000000G  EF  D6 00032 2$:   INCL    CRD$GL_CURRENT_GENERAL_COM        ; 0334
            50      02  D0 00038 3$:   MOVL    #2, R0           ; 0337
        04 0003B     RET                 ; 0338

; Routine Size:  60 bytes,    Routine Base:  CODE + 008C
```

```
;    0339  1 %SBTTL 'CRD$Assign_PTable_Ptrs'
;    0340  1
;    0341  1 GLOBAL ROUTINE CRD$Assign_PTable_Ptrs =
;    0342  1
;    0343  1 !++
;    0344  1 ! Functional Description:          [17]
;    0345  1 !
;    0346  1 ! This routine is run after the AutoSizer has attached all
;    0347  1 ! the devices and built the PTables.
;    0348  1 !   1. Determine Base System type and Hardware Support Table
;    0349  1 !   2. Build Device Data Blocks (DDB's)
;    0350  1 !   3. Put the PTable pointers into the Device Data Blocks.
;    0351  1 !
;    0352  1 ! Formal Parameters:
;    0353  1 !
;    0354  1 ! None
;    0355  1 !
;    0356  1 ! Side Effects:            [17]
;    0357  1 !
;    0358  1 ! CRD is stopped if Invalid Base System.
;    0359  1 !
;    0360  1 ! Completion Codes:
;    0361  1 !
;    0362  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0363  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0364  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0365  1 !--
```

```
;    0366  2 BEGIN     ! Routine CRD$Assign_PTable_Ptrs
;    0367  2  LOCAL                ![16]
;    0368  2      DDB_Ptr;
;    0369  2
;    0370  2  !+                   [17]
;    0371  2  ! Determine Base System and Hardware Support Table
;    0372  2  !-
;    0373  3  IF (CRD$Determine_Base_System () NEQ CRD$K_Success)
;    0374  2  THEN
;    0375  2      CRD$Stop (AUTO$K_Exit_Inv_Base_System);
;    0376  2
;    0377  2  !+                   [17]
;    0378  2  ! Build Queue of Device Data Blocks
;    0379  2  !-
;    0380  2  CRD$Build_DDBs ();
;    0381  2
;    0382  2  DDB_Ptr = .CRD$GA_First_Diagnostic; !        [16]
;    0383  2       !
;    0384  2       !          V
;    0385  2      ! Point to the first DDB
;    0386  2  DO
;    0387  3   BEGIN
;    0388  3   MAP
;    0389  3    DDB_Ptr : REF Device_Data_Block_Structure;
;    0390  3   !+
;    0391  3   ! Find a PTable entry for the device name in the Hardware Support table.
;    0392  3   ! Insert this pointer (or 0 if no PTable for the device) into the DDB.
;    0393  3   !-
;    0394  3
;    0395  3   DDB_Ptr [CRD$K_DDB_PTable_Entry] = CRD$Find_PTable (
;    0396  4    (
;    0397  4    BIND
;    0398  5     Diag_Support_Vector = (.DDB_Ptr [CRD$K_DDB_Auto_Support_Entry])
;    0399  4      : Hardware_Support_Vector;
;    0400  4
;    0401  4    .Diag_Support_Vector [CRD$K_Device_Name]
;    0402  4    )
;    0403  3   );
;    0404  3
;    0405  3   CRD$Print_DDB (.DDB_Ptr);
;    0406  3
;    0407  3   DDB_Ptr = .DDB_Ptr [CRD$K_DDB_FLink];
;    0408  3      ! Advance to the next DDB in the queue
;    0409  3   END
;    0410  2  UNTIL (.DDB_Ptr EQL .CRD$GA First_Diagnostic);        ! ^
;    0411  2                 ! |
;    0412  3  RETURN (CRD$K_Repeat_Turing) ! Repeat Turing_Machine again.      ![16]
;    0413  1 END;     ! Routine CRD$Assign_PTable_Ptrs


          .EXTRN  CRD$STOP, CRD$BUILD_DDBS
          .EXTRN  CRD$FIND_PTABLE

            0004 00000     .ENTRY  CRD$ASSIGN_PTABLE_PTRS, Save R2       ; 0341
```

```
ZZ-ENSAB-2.0    CRD$Assign_PTable_Ptrs                          19-Sep-1985      Fiche 2  Frame I13        Sequence 370
CRDACT2  *** CRDACT2 Module    19-Sep-1985 16:32:55 VAX-11 Bliss-32 V4.1-746     Page  20
01-17  CRD$Assign_PTable_Ptrs     3-Jan-1985 17:02:00 [DS.CRD.V020]CRDACT2.B32;1  (14)

       00000000V  EF       00  FB 00002    CALLS   #0, CRD$DETERMINE_BASE_SYSTEM        ; 0373
           01       50  D1 00009    CMPL    R0, #1          ;
           09  13 0000C    BEQL    1$              ;
           08  DD 0000E    PUSHL   #8              ; 0375
       00000000G  EF       01  FB 00010    CALLS   #1, CRD$STOP            ;
       00000000G  EF       00  FB 00017 1$:   CALLS   #0, CRD$BUILD_DDBS          ; 0380
           52 00000000G  EF  D0 0001E    MOVL    CRD$GA_FIRST_DIAGNOSTIC, DDB_PTR       ; 0382
           50 0C    A2  D0 00025 2$:   MOVL    12(DDB_PTR), R0          ; 0398
      0C   A0  DD 00029    PUSHL   12(R0)          ; 0401
       00000000G  EF       01  FB 0002C    CALLS   #1, CRD$FIND_PTABLE          ;
      08   A2       50  D0 00033    MOVL    R0, 8(DDB_PTR)          ;
           52  DD 00037    PUSHL   DDB_PTR         ; 0405
       00000000G  EF       01  FB 00039    CALLS   #1, CRD$PRINT_DDB          ;
           52       62  D0 00040    MOVL    (DDB_PTR), DDB_PTR          ; 0407
       00000000G  EF       52  D1 00043    CMPL    DDB_PTR, CRD$GA_FIRST_DIAGNOSTIC       ; 0410
           D9  12 0004A    BNEQ    2$              ;
           50       02  D0 0004C    MOVL    #2, R0          ; 0412
        04 0004F    RET                 ; 0413

; Routine Size:  80 bytes,    Routine Base:  CODE + 00C8
```

```
;    0414  1 %SBTTL 'CRD$AutoSizer_Set_Flags'
;    0415  1
;    0416  1 GLOBAL ROUTINE CRD$AutoSizer_Set_Flags (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0417  1
;    0418  1 !++
;    0419  1 ! Functional Description:
;    0420  1 !
;    0421  1 ! Load the AutoSizer's SET FLAGS command into the CLI buffer. Note: This is a mirror image of the
;    0422  1 ! Load_Set_Flags_Com routine above. This could be eliminated, but it would make the state table
;    0423  1 ! and make this routine more complicated.
;    0424  1 !
;    0425  1 ! Formal Parameters:
;    0426  1 !
;    0427  1 ! CLI_Buffer_Length  : The length of the CLI buffer.
;    0428  1 ! CLI_Buffer_Address : The address of the CLI buffer.
;    0429  1 !
;    0430  1 ! Side Effects:
;    0431  1 !
;    0432  1 ! None
;    0433  1 !
;    0434  1 ! Completion Codes:
;    0435  1 !
;    0436  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0437  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0438  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0439  1 !--
```

```
;    0440  2 BEGIN     ! Routine CRD$AutoSizer_Set_Flags
;    0441  2  MAP
;    0442  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0443  2
;    0444  2  MAP
;    0445  2   CRD$GT_DS_SET_FLAGS_Com : VECTOR [, BYTE];
;    0446  2
;    0447  2  MAP              ![13]
;    0448  2   CRD$GT_AutoSizer_Set_Flags : VECTOR [, BYTE];       ![13]
;    0449  2
;    0450  2  IF (.CRD$GT_AutoSizer_Set_Flags [0] EQL 0) THEN        ![13]
;    0451  3   RETURN (CRD$K_Repeat_Turing)
;    0452  2  ELSE
;    0453  2   CH$COPY (
;    0454  2    .CRD$GT_DS_SET_FLAGS_Com [0],    CH$PTR (CRD$GT_DS_SET_FLAGS_Com [1]),   ![13]
;    0455  2    .CRD$GT_AutoSizer_Set_Flags [0], CH$PTR (CRD$GT_AutoSizer_Set_Flags [1]),  ![13]
;    0456  2    %C' ',
;    0457  2    .CLI_Buffer_Length,       CH$PTR (.CLI_Buffer_Address)
;    0458  2    );
;    0459  2
;    0460  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0461  2
;    0462  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0463  1 END;     ! Routine CRD$AutoSizer_Set_Flags



          .EXTRN  CRD$GT_DS_SET_FLAGS_COM
          .EXTRN  CRD$GT_AUTOSIZER_SET_FLAGS
          .EXTRN  CRD$PRINT_DS_COMMAND

            03FC 00000     .ENTRY  CRD$AUTOSIZER_SET_FLAGS, Save R2,R3,R4,R5,- ; 0416
           R6,R7,R8,R9             ;
          59 00000000G  EF  9A 00002     MOVZBL  CRD$GT_AUTOSIZER_SET_FLAGS, R9       ; 0450
          04  12 00009     BNEQ    1$              ;
          50       02  D0 0000B     MOVL    #2, R0          ; 0451
       04 0000E     RET                     ;
          58 00000000G  EF  9A 0000F 1$:    MOVZBL  CRD$GT_DS_SET_FLAGS_COM, R8        ; 0454
          57 04    AC  D0 00016     MOVL    CLI_BUFFER_LENGTH, R7          ; 0457
          56 08    AC  D0 0001A     MOVL    CLI_BUFFER_ADDRESS, R6         ;
    57          20 00000000G  EF       58  2C 0001E     MOVC5   R8, CRD$GT_DS_SET_FLAGS_COM+1, #32, R7, -    ;
          66       00027     (R6)            ;
          10  18 00028     BGEQ    2$              ;
          56       58  C0 0002A     ADDL2   R8, R6          ;
          57       58  C2 0002D     SUBL2   R8, R7          ;
    57          20 00000000G  EF       59  2C 00030     MOVC5   R9, CRD$GT_AUTOSIZER_SET_FLAGS+1, #32, R7, -;
          66       00039     (R6)            ;
          7E 04    AC  7D 0003A 2$:    MOVQ    CLI_BUFFER_LENGTH, -(SP)         ; 0460
       00000000G  EF       02  FB 0003E     CALLS   #2, CRD$PRINT_DS_COMMAND         ;
          50 00500001    8F  D0 00045     MOVL    #5242881, R0          ; 0462
       04 0004C     RET                     ; 0463

; Routine Size:  77 bytes,    Routine Base:  CODE + 0118
```

```
;    0464  1
;    0465  1
;    0466  1
```

```
;    0467  1 %SBTTL 'CRD$Determine_Base_System'
;    0468  1       !          [17]
;    0469  1
;    0470  1 GLOBAL ROUTINE CRD$Determine_Base_System  =
;    0471  1
;    0472  1 !++
;    0473  1 ! Functional Description:
;    0474  1 !
;    0475  1 !    Determines if a valid base system exists.  The system is
;    0476  1 !    determined by the folowing algorithm for the specified
;    0477  1 !    hardware configurations:
;    0478  1 !
;    0479  1 !     Configuration     ! System
;    0480  1 ! ----------------------!
;    0481  1 ! IDC KLESI UDA50 !
;    0482  1 ! --- ---- ----- ! ------
;    0483  1 ! 1 0 0  IDC
;    0484  1 ! 0 1 0  LESI
;    0485  1 ! 0 0 1  UDA50
;    0486  1 ! 1 1 0  IDC
;    0487  1 ! 1 0 1  IDC
;    0488  1 ! 1 1 1  IDC
;    0489  1 ! 0 1 1  LESI or UDA50
;    0490  1 !    whichever is at CSR=772150
;    0491  1 ! 0 0 0  Failure
;    0492  1 !
;    0493  1
;    0494  1 !
;    0495  1 ! Formal Parameters:
;    0496  1 !
;    0497  1 ! none
;    0498  1 !
;    0499  1 ! Implicit Outputs
;    0500  1 !
;    0501  1 ! CRD$GB_Base_System is specified with correct code.  Initialized
;    0502  1 ! if failure.
;    0503  1 ! CRD$GB_Hdwr_Sprt_Table is specified with correct table pointer.
;    0504  1 ! Initialized if failure.
;    0505  1 !
;    0506  1 ! Side Effects:
;    0507  1 !
;    0508  1 ! None
;    0509  1 !
;    0510  1 ! Completion Codes:
;    0511  1 !
;    0512  1 ! CRD$K_Failure => Failure.  Invalid or No Base System Found.
;    0513  1 ! CRD$K_Success => Success.  Valid base system found.
;    0514  1 !--
```

```
;    0515  2 BEGIN     ! Routine CRD$Determine_Base_System
;    0516  2 BIND
;    0517  2  IDC_Device_Name  = $ASCIC ('DQA'),
;    0518  2  LESI_Device_Name = $ASCIC ('DAA'),
;    0519  2  UDA_Device_Name = $ASCIC ('DUA'),
;    0520  2  RA60_Drive_0_Device_Name
;    0521  2     = $ASCIC ('DJA0'),
;    0522  2  RA8081_Drive_0_Device_Name
;    0523  2     = $ASCIC ('DUA0'),
;    0524  2  RA60_Drive_1_Device_Name
;    0525  2     = $ASCIC ('DJA1');
;    0526  2 REGISTER
;    0527  2  IDC_PTable : LONG INITIAL (False),
;    0528  2  LESI_PTable : LONG INITIAL (False),
;    0529  2  UDA_PTable : LONG INITIAL (False),
;    0530  2  RA60_Drive_0_PTable
;    0531  2     : LONG INITIAL (False),
;    0532  2  RA8081_Drive_0_PTable
;    0533  2     : LONG INITIAL (False),
;    0534  2  RA60_Drive_1_PTable
;    0535  2     : LONG INITIAL (False);
;    0536  2 LOCAL
;    0537  2  CSR  : WORD,
;    0538  2  PTable_Ptr : REF $DS_HPO_DECL();
;    0539  2
;    0540  2 CRD$GB_Base_System = CRD$K_Base_System_Null;
;    0541  2 CRD$GA_Hdwr_Sprt_Table = 0;
;    0542  2
;    0543  2 IDC_PTable = CRD$Find_PTable (IDC_Device_Name);
;    0544  2 LESI_PTable = CRD$Find_PTable (LESI_Device_Name);
;    0545  2 UDA_PTable = CRD$Find_PTable (UDA_Device_Name);
;    0546  2 RA60_Drive_0_PTable = CRD$Find_PTable (RA60_Drive_0_Device_Name);
;    0547  2 RA8081_Drive_0_PTable = CRD$Find_PTable (RA8081_Drive_0_Device_Name);
;    0548  2 RA60_Drive_1_PTable = CRD$Find_PTable (RA60_Drive_1_Device_Name);
;    0549  2
;    0550  2
;    0551  2 IF .IDC_PTable NEQ 0
;    0552  2 THEN
;    0553  3     BEGIN
;    0554  3     CRD$GB_Base_System = CRD$K_Base_System_IDC;
;    0555  3     CRD$GA_Hdwr_Sprt_Table = CRD$AL_IDC_Hdwr_Sprt_Table;
;    0556  3     RETURN CRD$K_Success;
;    0557  2     END;
;    0558  2
;    0559  2 IF .LESI_PTable NEQ 0
;    0560  2 THEN
;    0561  3     BEGIN
;    0562  3     PTable_Ptr = .LESI_PTable;
;    0563  3     CSR = .PTable_Ptr [HP$A_Device];
;    0564  3     IF .CSR EQL %O'172150'
;    0565  3     THEN
;    0566  4  BEGIN
;    0567  4  CRD$GB_Base_System = CRD$K_Base_System_LESI;
;    0568  4  CRD$GA_Hdwr_Sprt_Table = CRD$AL_LESI_Hdwr_Sprt_Table;
;    0569  4  RETURN CRD$K Success;
```

```
;    0570  3  END;
;    0571  2     END;
;    0572  2
;    0573  2 IF .UDA_PTable NEQ 0
;    0574  2 THEN
;    0575  3     BEGIN
;    0576  3     PTable_Ptr = .UDA_PTable;
;    0577  3     CSR = .PTable_Ptr [HP$A_Device];
;    0578  3     IF .CSR EQL x0'172150'
;    0579  3     THEN
;    0580  4  BEGIN
;    0581  4  !
;    0582  4  ! Drive 1 - RA60
;    0583  4  ! Drive 0 - RA60
;    0584  4  !
;    0585  5  IF (.RA60_Drive_1_PTable NEQ 0) AND (.RA60_Drive_0_PTable NEQ 0)
;    0586  4  THEN
;    0587  5      BEGIN
;    0588  5      CRD$GB_Base_System = CRD$K_Base_System_UDA_3;
;    0589  5      CRD$GA_Hdwr_Sprt_Table = CRD$AL_UDA_3_Hdwr_Sprt_Table;
;    0590  5      RETURN CRD$K_Success;
;    0591  4      END;
;    0592  4  !
;    0593  4  ! Drive 1 - RA60
;    0594  4  ! Drive 0 - RA80, 81, or Nothing
;    0595  4  !
;    0596  4  IF (.RA60_Drive_1_PTable NEQ 0) AND
;    0597  5     ( (.RA8081_Drive_0_PTable NEQ 0) OR (.RA8081_Drive_0_PTable EQL 0) )
;    0598  4  THEN
;    0599  5      BEGIN
;    0600  5      CRD$GB_Base_System = CRD$K_Base_System UDA 1;
;    0601  5      CRD$GA_Hdwr_Sprt_Table = CRD$AL_UDA_1_Hdwr Sprt_Table;
;    0602  5      RETURN CRD$K_Success;
;    0603  4      END;
;    0604  4  !
;    0605  4  ! Drive 1 - no RA60
;    0606  4  ! Drive 0 - RA80 or 81
;    0607  4  !
;    0608  5  IF (.RA60_Drive_1_PTable EQL 0) AND (.RA8081_Drive_0_PTable NEQ 0)
;    0609  4  THEN
;    0610  5      BEGIN
;    0611  5      CRD$GB_Base System = CRD$K_Base_System_UDA_0;
;    0612  5      CRD$GA_Hdwr_Sprt_Table = CRD$AL_UDA_0_Hdwr_Sprt_Table;
;    0613  5      RETURN CRD$K_Success;
;    0614  4      END;
;    0615  4  !
;    0616  4  ! Drive 1    no RA60
;    0617  4  ! Drive 0 - RA60
;    0618  4  !
;    0619  5  IF (.RA60_Drive 1_PTab e EQL 0) AND (.RA60_Drive_0_PTable NEQ 0)
;    0620  4  THEN
;    0621  5      BEGIN
;    0622  5      CRD$GB_Base_System = CRD$K_Base_System_UDA_2;
;    0623  5      CRD$GA_Hdwr Sprt Table = CRD$AL_UDA_2_Hdwr_Sprt_Table;
;    0624  5      RETURN CRD$K_Success;
```

```
;   0625  4      END;
;   0626  4  !
;   0627  4  ! Default:
;   0628  4  ! Drive 1 - no RA60
;   0629  4  ! Drive 0 - no RA60, 80, or 81
;   0630  4  !
;   0631  4  CRD$GB_Base_System = CRD$K_Base_System_UDA_0;
;   0632  4  CRD$GA_Hdwr_Sprt_Table = CRD$AL_UDA_0_Hdwr_Sprt_Table;
;   0633  4  RETURN CRD$K_Success;
;   0634  3  END;
;   0635  2     END;
;   0636  2
;   0637  2 RETURN CRD$K_Failure;
;   0638  1 END;


          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

        41  51  44  03  0007F P.AAE:  .ASCII  <3>\DQA\          ;
        41  41  44  03  00083 P.AAF:  .ASCII  <3>\DAA\          ;
        41  55  44  03  00087 P.AAG:  .ASCII  <3>\DUA\          ;
     30 41  4A  44  04  0008B P.AAH:  .ASCII  <4>\DJA0\          ;
     30 41  55  44  04  00090 P.AAI:  .ASCII  <4>\DUA0\          ;
     31 41  4A  44  04  00095 P.AAJ:  .ASCII  <4>\DJA1\          ;

         IDC_DEVICE_NAME=    P.AAE
         LESI_DEVICE_NAME=   P.AAF
         UDA_DEVICE_NAME=    P.AAG
         RA60_DRIVE_0_DEVICE_NAME=
               P.AAH
         RA8081_DRIVE_0_DEVICE_NAME=
               P.AAI
         RA60_DRIVE_1_DEVICE_NAME=
               P.AAJ
          .EXTRN  CRD$GB_BASE_SYSTEM
          .EXTRN  CRD$GA_HDWR_SPRT_TABLE
          .EXTRN  CRD$AL_IDC_HDWR_SPRT_TABLE
          .EXTRN  CRD$AL_LESI_HDWR_SPRT_TABLE
          .EXTRN  CRD$AL_UDA_3_HDWR_SPRT_TABLE
          .EXTRN  CRD$AL_UDA_1_HDWR_SPRT_TABLE
          .EXTRN  CRD$AL_UDA_0_HDWR_SPRT_TABLE
          .EXTRN  CRD$AL_UDA_2_HDWR_SPRT_TABLE

          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            00FC 00000     .ENTRY  CRD$DETERMINE_BASE_SYSTEM, Save R2,R3,R4,-   ; 0470
           R5,R6,R7              ;
          52  7C 00002     CLRQ    IDC_PTABLE           ; 0515
          54  7C 00004     CLRQ    UDA_PTABLE           ;
          56  7C 00006     CLRQ    RA8081_DRIVE_0_PTABLE          ;
00000000G   EF  94 00008     CLRB    CRD$GB_BASE_SYSTEM          ; 0540
00000000G   EF  D4 0000E     CLRL    CRD$GA_HDWR_SPRT_TABLE          ; 0541
00000000'   EF  9F 00014     PUSHAB  IDC_DEVICE_NAME          ; 0543
00000000G   EF       01  FB 0001A     CALLS   #1, CRD$FIND_PTABLE          ;
```

```
                52      50  D0 00021     MOVL    R0, IDC_PTABLE          ;
          00000000'  EF  9F 00024     PUSHAB  LESI_DEVICE_NAME        ; 0544
          00000000G  EF      01  FB 0002A     CALLS   #1, CRD$FIND_PTABLE         ;
                53      50  D0 00031     MOVL    R0, LESI_PTABLE         ;
          00000000'  EF  9F 00034     PUSHAB  UDA_DEVICE_NAME         ; 0545
          00000000G  EF      01  FB 0003A     CALLS   #1, CRD$FIND_PTABLE         ;
                54      50  D0 00041     MOVL    R0, UDA_PTABLE          ;
          00000000'  EF  9F 00044     PUSHAB  RA60_DRIVE_0_DEVICE_NAME        ; 0546
          00000000G  EF      01  FB 0004A     CALLS   #1, CRD$FIND_PTABLE         ;
                55      50  D0 00051     MOVL    R0, RA60_DRIVE_0_PTABLE         ;
          00000000'  EF  9F 00054     PUSHAB  RA8081_DRIVE_0_DEVICE_NAME      ; 0547
          00000000G  EF      01  FB 0005A     CALLS   #1, CRD$FIND_PTABLE         ;
                56      50  D0 00061     MOVL    R0, RA8081_DRIVE_0_PTABLE       ;
          00000000'  EF  9F 00064     PUSHAB  RA60_DRIVE_1_DEVICE_NAME        ; 0548
          00000000G  EF      01  FB 0006A     CALLS   #1, CRD$FIND_PTABLE         ;
                57      50  D0 00071     MOVL    R0, RA60_DRIVE_1_PTABLE         ;
                52  D5 00074     TSTL    IDC_PTABLE              ; 0551
                15  13 00076     BEQL    1$              ;
          00000000G  EF      01  90 00078     MOVB    #1, CRD$GB_BASE_SYSTEM          ; 0554
          00000000G  EF 00000000G  EF  9E 0007F     MOVAB   CRD$AL_IDC_HDWR_SPRT_TABLE, -        ; 0555
               CRD$GA_HDWR_SPRT_TABLE         ;
              00AE  31 00083     BRW     8$              ; 0556
                53  D5 00086 1$:   TSTL    LESI_PTABLE             ; 0559
                23  13 00088     BEQL    2$              ;
                50      53  D0 0008A     MOVL    LESI_PTABLE, PTABLE_PTR         ; 0562
                51  18      A0  B0 0008D     MOVW    24(PTABLE_PTR), CSR         ; 0563
            F468    8F      51  B1 00091     CMPW    CSR, #62568             ; 0564
                15  12 00096     BNEQ    2$              ;
          00000000G  EF      02  90 00098     MOVB    #2, CRD$GB_BASE_SYSTEM          ; 0567
          00000000G  EF 00000000G  EF  9E 0009F     MOVAB   CRD$AL_LESI_HDWR_SPRT_TABLE, -        ; 0568
               CRD$GA_HDWR_SPRT_TABLE         ;
              0087  31 000AA     BRW     8$              ; 0569
                54  D5 000AD 2$:   TSTL    UDA_PTABLE              ; 0573
                03  12 000AF     BNEQ    3$              ;
              0084  31 000B1     BRW     9$              ;
                50      54  D0 000B4 3$:   MOVL    UDA_PTABLE, PTABLE_PTR          ; 0576
                51  18      A0  B0 000B7     MOVW    24(PTABLE_PTR), CSR         ; 0577
            F468    8F      51  B1 000BB     CMPW    CSR, #62568             ; 0578
                76  12 000C0     BNEQ    9$              ;
                50  D4 000C2     CLRL    R0              ; 0585
                57  D5 000C4     TSTL    RA60_DRIVE_1_PTABLE             ;
                1A  13 000C6     BEQL    4$              ;
                50  D6 000C8     INCL    R0              ;
                55  D5 000CA     TSTL    RA60_DRIVE_0_PTABLE             ;
                14  13 000CC     BEQL    4$              ;
          00000000G  EF      06  90 000CE     MOVB    #6, CRD$GB_BASE_SYSTEM          ; 0588
          00000000G  EF 00000000G  EF  9E 000D5     MOVAB   CRD$AL_UDA_3_HDWR_SPRT_TABLE, -        ; 0589
               CRD$GA_HDWR_SPRT_TABLE         ;
                52  11 000E0     BRB     8$              ; 0590
                16      50  E9 000E2 4$:   BLBC    R0, 5$              ; 0596
                56  D5 000E5     TSTL    RA8081_DRIVE_0_PTABLE           ; 0597
          00000000G  EF      04  90 000E7     MOVB    #4, CRD$GB_BASE_SYSTEM          ; 0600
          00000000G  EF 00000000G  EF  9E 000EE     MOVAB   CRD$AL_UDA_1_HDWR_SPRT_TABLE, -        ; 0601
               CRD$GA_HDWR_SPRT_TABLE         ;
                39  11 000F9     BRB     8$              ; 0602
```

```
            50  D4 00102 5$:   CLRL    R0              ; 0608
            57  D5 00104     TSTL    RA60_DRIVE_1_PTABLE          ;
            06  12 00106     BNEQ    6$              ;
            50  D6 00108     INCL    R0              ;
            56  D5 0010A     TSTL    RA8081_DRIVE_0_PTABLE          ;
            1B  12 0010C     BNEQ    7$              ;
            18      50  E9 0010E 6$:   BLBC    R0, 7$          ; 0619
            55  D5 00111     TSTL    RA60_DRIVE_0_PTABLE          ;
            14  13 00113     BEQL    7$              ;
       00000000G  EF      05  90 00115     MOVB    #5, CRD$GB_BASE_SYSTEM          ; 0622
       00000000G  EF 00000000G  EF  9E 0011C     MOVAB   CRD$AL_UDA_2_HDWR_SPRT_TABLE, -       ; 0623
            CRD$GA_HDWR_SPRT_TABLE          ;
            12  11 00127     BRB     8$              ; 0624
       00000000G  EF      03  90 00129 7$:   MOVB    #3, CRD$GB_BASE_SYSTEM          ; 0631
       00000000G  EF 00000000G  EF  9E 00130     MOVAB   CRD$AL_UDA_0_HDWR_SPRT_TABLE, -       ; 0632
            CRD$GA_HDWR_SPRT_TABLE          ;
            50      01  D0 0013B 8$:   MOVL    #1, R0          ; 0633
         04 0013E     RET                ;
            50  D4 0013F 9$:   CLRL    R0              ; 0637
         04 00141     RET                ; 0638

; Routine Size:  322 bytes,    Routine Base:  CODE + 0165
```

```
;    0639  1 %SBTTL 'CRD$Done_General_Coms'
;    0640  1
;    0641  1 GLOBAL ROUTINE CRD$Done_General_Coms =
;    0642  1
;    0643  1 !++
;    0644  1 ! Functional Description:
;    0645  1 !
;    0646  1 ! Do anything here that needs to be done before the next set of DS commands is done.
;    0647  1 !
;    0648  1 ! Formal Parameters:
;    0649  1 !
;    0650  1 ! None
;    0651  1 !
;    0652  1 ! Side Effects:
;    0653  1 !
;    0654  1 ! None
;    0655  1 !
;    0656  1 ! Completion Codes:
;    0657  1 !
;    0658  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0659  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0660  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0661  1 !--
```

```
;    0662  2 BEGIN     ! Routine CRD$Done_General_Coms
;    0663  2  IF (.CRD$GB_CRD_Debug) THEN
; P  0664  2   $CRD_Print ($ASCIC ('*** Done_General_Coms: Current = !XL!/'),
;    0665  2    .CRD$GL_Current_General_Com);
;    0666  2
;    0667  2  CRD$GL_Current_General_Com = 0; ! Reset the General_Com pointer
;    0668  3  RETURN (CRD$K_Repeat_Turing)
;    0669  3
;    0670  1 END;     ! Routine CRD$Done_General_Coms



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

72  65  6E  65  47  5F  65  6E  6F  44  20  2A  2A  2A  26  0009A P.AAK:  .ASCII  \&*** Done_General_Coms: Current = !XL!/\   ;
6E  65  72  72  75  43  20  3A  73  6D  6F  43  5F  6C  61  000A9             ;
    2F  21  4C  58  21  20  3D  20  74  000B8             ;



            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0000 00000     .ENTRY  CRD$DONE_GENERAL_COMS, Save nothing       ; 0641
            17 00000000G  EF  E9 00002     BLBC    CRD$GB_CRD_DEBUG, 1$         ; 0663
         00000000G  EF  DD 00009     PUSHL   CRD$GL_CURRENT_GENERAL_COM       ; 0665
         00000000'  EF  9F 0000F     PUSHAB  P.AAK            ;
             01  DD 00015     PUSHL   #1             ;
             1A  DD 00017     PUSHL   #26             ;
         00000000G  FF       04  FB 00019     CALLS   #4, aCRD$PRINT          ;
         00000000G  EF  D4 00020 1$:   CLRL    CRD$GL_CURRENT_GENERAL_COM       ; 0667
              50       02  D0 00026     MOVL    #2, R0           ; 0668
           04 00029     RET                 ; 0670

; Routine Size:  42 bytes,    Routine Base:  CODE + 02A7
```

```
;    0671  1 %SBTTL 'CRD$Level_4_Passed'
;    0672  1
;    0673  1 GLOBAL ROUTINE CRD$Level_4_Passed =
;    0674  1
;    0675  1 !++
;    0676  1 ! Functional Description:
;    0677  1 !
;    0678  1 ! Print PASSES for the level 4 diagnostic that ran previous to the Macro diagnostics.
;    0679  1 !
;    0680  1 ! Formal Parameters:
;    0681  1 !
;    0682  1 ! None
;    0683  1 !
;    0684  1 ! Side Effects:
;    0685  1 !
;    0686  1 ! None
;    0687  1 !
;    0688  1 ! Completion Codes:
;    0689  1 !
;    0690  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0691  1 !--
```

```
;   0692  2 BEGIN     ! Routine CRD$Level_4_Passed
;   0693  2  $CRD_Print (CRD$GT_Pass);
;   0694  2
;   0695  2  RETURN (CRD$K_Repeat_Turing);
;   0696  1 END;      ! Routine CRD$Level_4_Passed




                0000 00000     .ENTRY  CRD$LEVEL_4_PASSED, Save nothing       ; 0673
            00000000G  EF  9F 00002     PUSHAB  CRD$GT_PASS           ; 0693
                01  DD 00008     PUSHL   #1            ;
                1A  DD 0000A     PUSHL   #26           ;
            00000000G  FF       03  FB 0000C     CALLS   #3, @CRD$PRINT        ;
                50       02  D0 00013     MOVL    #2, R0           ; 0695
             04 00016     RET                 ; 0696

; Routine Size:  23 bytes,    Routine Base:  CODE + 02D1
```

```
;    0697  1 %SBTTL 'CRD$Load_AutoSizer'
;    0698  1
;    0699  1 GLOBAL ROUTINE CRD$Load_AutoSizer (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0700  1
;    0701  1 !++
;    0702  1 ! Functional Description:
;    0703  1 !
;    0704  1 ! Load the LOAD EVSBA command into the CLI buffer.
;    0705  1 !
;    0706  1 ! Formal Parameters:
;    0707  1 !
;    0708  1 ! CLI_Buffer_Length  : The length of the CLI buffer.
;    0709  1 ! CLI_Buffer_Address : The address of the CLI buffer.
;    0710  1 !
;    0711  1 ! Side Effects:
;    0712  1 !
;    0713  1 ! None
;    0714  1 !
;    0715  1 ! Completion Codes:
;    0716  1 !
;    0717  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0718  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0719  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0720  1 !--
```

```
;    0721  2 BEGIN     ! Routine CRD$Load_AutoSizer
;    0722  2  MAP
;    0723  2   CRD$GT_DS_LOAD_Com : VECTOR [, BYTE];
;    0724  2
;    0725  2  MAP              ![13]
;    0726  2   CRD$GT_AutoSizer_Name : VECTOR [, BYTE];       ![13]
;    0727  2
;    0728  2  CH$COPY (            ![13]
;    0729  2    .CRD$GT_DS_LOAD_Com [0],    CH$PTR (CRD$GT_DS_LOAD_Com [1]),     ![13]
;    0730  2    .CRD$GT_AutoSizer_Name [0], CH$PTR (CRD$GT_AutoSizer_Name [1]),     ![13]
;    0731  2    %C' ',
;    0732  2    .CLI_Buffer_Length,  CH$PTR (.CLI_Buffer_Address)
;    0733  2    );
;    0734  2
;    0735  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0736  2
;    0737  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0738  1 END;     ! Routine CRD$Load_AutoSizer



          .EXTRN  CRD$GT_DS_LOAD_COM
          .EXTRN  CRD$GT_AUTOSIZER_NAME

            03FC 00000     .ENTRY  CRD$LOAD_AUTOSIZER, Save R2,R3,R4,R5,R6,R7,-; 0699
           R8,R9            ;
          59 00000000G  EF  9A 00002     MOVZBL  CRD$GT_DS_LOAD_COM, R9         ; 0729
          58 00000000G  EF  9A 00009     MOVZBL  CRD$GT_AUTOSIZER_NAME, R8      ; 0730
          57 04    AC  D0 00010     MOVL    CLI_BUFFER_LENGTH, R7         ; 0732
          56 08    AC  D0 00014     MOVL    CLI_BUFFER_ADDRESS, R6        ;
    57          20 00000000G  EF       59  2C 00018     MOVC5   R9, CRD$GT_DS_LOAD_COM+1, #32, R7, (R6)     ;
          66       00021            ;
          10  18 00022     BGEQ    1$            ;
          56       59  C0 00024     ADDL2   R9, R6          ;
          57       59  C2 00027     SUBL2   R9, R7          ;
    57          20 00000000G  EF       58  2C 0002A     MOVC5   R8, CRD$GT_AUTOSIZER_NAME+1, #32, R7, (R6)  ;
          66       00033            ;
          7E 04    AC  7D 00034 1$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)        ; 0735
       00000000G  EF       02  FB 00038     CALLS   #2, CRD$PRINT_DS_COMMAND       ;
          50 00500001    8F  D0 0003F     MOVL    #5242881, R0           ; 0737
           04 00046     RET                 ; 0738

; Routine Size:  71 bytes,    Routine Base:  CODE + 02E8
```

```
;    0739  1 %SBTTL 'CRD$Load_General_Com'
;    0740  1
;    0741  1 GLOBAL ROUTINE CRD$Load_General_Com (Data_Max_Length, Data_Buffer_Address) =
;    0742  1
;    0743  1 !++
;    0744  1 ! Functional Description:
;    0745  1 !
;    0746  1 ! Load a general command into the CLI data buffer.
;    0747  1 !
;    0748  1 ! Formal Parameters:
;    0749  1 !
;    0750  1 ! Data_Max_Length     : The maximum length of the data to insert into the buffer.
;    0751  1 ! Data_Buffer_Address : The buffer where the data is to be inserted.
;    0752  1 !
;    0753  1 ! Side Effects:
;    0754  1 !
;    0755  1 ! None
;    0756  1 !
;    0757  1 ! Completion Codes:
;    0758  1 !
;    0759  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0760  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0761  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0762  1 !--
```

```
;    0763  2 BEGIN     ! Routine CRD$Load_General_Com
;    0764  2  LOCAL
;    0765  2   Command_Ptr;
;    0766  2
;    0767  2  BIND
;    0768  2   Command_Output    = $ASCIC ('The DS command now executing is: .!AD.!/');
;    0769  2
;    0770  2  IF ((.Data_Max_Length LSS 1) OR (CRD$K_DS_Command_Size GTR .Data_Max_Length)) THEN
;    0771  3   BEGIN
;    0772  3   CRD$Internal_Error (CRD$K_Data_Length_Error, .Data_Max_Length, .Data_Buffer_Address);  ![02]
;    0773  4   RETURN (CRD$K_Failure)
;    0774  2   END;
;    0775  2
;    0776  2  Command_Ptr = CRD$Get_General_Command ();
;    0777  2      ! Get a pointer to the next command
;    0778  2
;    0779  2  CH$MOVE (CRD$K_DS_Command_Size, .Command_Ptr, .Data_Buffer_Address);
;    0780  2      ! Move the new command to the buffer
;    0781  2
;    0782  2  CRD$Print_DS_Command (.Data_Max_Length, .Data_Buffer_Address);
;    0783  2
;    0784  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0785  2      ! Return the size in the high word, success in the low word
;    0786  2
;    0787  1 END;     ! Routine CRD$Load_General_Com


          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

64  6E  61  6D  6D  6F  63  20  53  44  20  65  68  54  28  000C1 P.AAL:  .ASCII  \(The DS command now executing is: .!AD.!\  ;
20  67  6E  69  74  75  63  65  78  65  20  77  6F  6E  20  000D0            ;
      21  2E  44  41  21  2E  20  3A  73  69  000DF            ;
       2F  000E9    .ASCII  \/\           ;

         COMMAND_OUTPUT=     P.AAL
          .EXTRN  CRD$INTERNAL_ERROR
          .EXTRN  CRD$GET_GENERAL_COMMAND

          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            003C 00000    .ENTRY  CRD$LOAD_GENERAL_COM, Save R2,R3,R4,R5      ; 0741
     04    AC  D5 00002    TSTL    DATA_MAX_LENGTH          ; 0770
           0A  15 00005    BLEQ    1$          ;
      00000050   8F 04   AC  D1 00007    CMPL    DATA_MAX_LENGTH, #80          ;
           10  18 0000F    BGEQ    2$          ;
           7E 04   AC  7D 00011 1$:   MOVQ    DATA_MAX_LENGTH, -(SP)        ; 0772
           7E      05  CE 00015    MNEGL   #5, -(SP)          ;
      00000000G  EF       03  FB 00018    CALLS   #3, CRD$INTERNAL_ERROR        ;
           21  11 0001F    BRB     3$          ; 0773
      00000000G  EF       00  FB 00021 2$:   CALLS   #0, CRD$GET_GENERAL_COMMAND        ; 0776
     08   BC         60     0050   8F  28 00028    MOVC3   #80, (COMMAND_PTR), @DATA_BUFFER_ADDRESS    ; 0779
           7E 04   AC  7D 0002F    MOVQ    DATA_MAX_LENGTH, -(SP)        ; 0782
      00000000G  EF       02  FB 00033    CALLS   #2, CRD$PRINT_DS_COMMAND        ;
           50 00500001   8F  D0 0003A    MOVL    #5242881, R0          ; 0784
```

```
        04 00041     RET                ;
           50  D4 00042 3$:   CLRL     R0              ; 0787
        04 00044     RET                ;

; Routine Size:  69 bytes,    Routine Base:  CODE + 032F
```

```
;    0788  1 %SBTTL 'CRD$Load_Load_Com'
;    0789  1
;    0790  1 GLOBAL ROUTINE CRD$Load_Load_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0791  1
;    0792  1 !++
;    0793  1 ! Functional Description:
;    0794  1 !
;    0795  1 ! Load the next LOAD command into the CLI buffer.
;    0796  1 !
;    0797  1 ! Formal Parameters:
;    0798  1 !
;    0799  1 ! CLI_Buffer_Length  : The length of the CLI buffer.
;    0800  1 ! CLI_Buffer_Address : The address of the CLI buffer.
;    0801  1 !
;    0802  1 ! Side Effects:
;    0803  1 !
;    0804  1 ! None
;    0805  1 !
;    0806  1 ! Completion Codes:
;    0807  1 !
;    0808  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0809  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0810  1 ! CRD$K_Repeat_Turing  -> Cause the Turing_Machine routine to call itself recursively.
;    0811  1 !--
```

```
;    0812  2 BEGIN     ! Routine CRD$Load_Load_Com
;    0813  2  MAP
;    0814  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0815  2
;    0816  2  MAP
;    0817  2   CRD$GT_DS_LOAD_Com : VECTOR [, BYTE];
;    0818  2
;    0819  2  BIND
;    0820  3   Diag_Vector_Ptr = (.CRD$GA_Current_Diagnostic [CRD$K_DDB_Auto_Support_Entry])
;    0821  2    : Hardware_Support_Vector;
;    0822  2
;    0823  2  BIND
;    0824  2   Diag_Name = (.Diag_Vector_Ptr [CRD$K_Diagnostic_Name]) : VECTOR [, BYTE];
;    0825  2
;    0826  2  CRD$Print_DDB (.CRD$GA_Current_Diagnostic);
;    0827  2
;    0828  2  CH$COPY (
;    0829  2   .CRD$GT_DS_LOAD_Com [0], CH$PTR (CRD$GT_DS_LOAD_Com [1]),
;    0830  2   .Diag_Name [0],   CH$PTR (Diag_Name [1]),
;    0831  2   %C' ',
;    0832  2   .CLI_Buffer_Length,  CH$PTR (.CLI_Buffer_Address)
;    0833  2   );
;    0834  2
;    0835  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0836  2
;    0837  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0838  1 END;     ! Routine CRD$Load_Load_Com



          07FC 00000     .ENTRY  CRD$LOAD_LOAD_COM, Save R2,R3,R4,R5,R6,R7,- ; 0790
         R8,R9,R10           ;
        50 00000000G  EF  D0 00002     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0820
        58 0C    B0  D0 00009     MOVL    a12(R0), R8          ; 0824
        50  DD 0000D     PUSHL   R0           ; 0826
    00000000G  EF       01  FB 0000F     CALLS   #1, CRD$PRINT_DDB          ;
        5A 00000000G  EF  9A 00016     MOVZBL  CRD$GT_DS_LOAD_COM, R10        ; 0829
        59       68  9A 0001D     MOVZBL  (R8), R9          ; 0830
        57 04    AC  D0 00020     MOVL    CLI_BUFFER_LENGTH, R7          ; 0832
        56 08    AC  D0 00024     MOVL    CLI_BUFFER_ADDRESS, R6         ;
   57          20 00000000G  EF       5A  2C 00028     MOVC5   R10, CRD$GT_DS_LOAD_COM+1, #32, R7, (R6)     ;
        66      00031           ;
        0D  18 00032     BGEQ    1$           ;
        56       5A  C0 00034     ADDL2   R10, R6          ;
        57       5A  C2 00037     SUBL2   R10, R7          ;
   57          20    01    A8       59  2C 0003A     MOVC5   R9, 1(R8), #32, R7, (R6)         ;
        66      00040           ;
        7E 04    AC  7D 00041 1$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)         ; 0835
    00000000G  EF       02  FB 00045     CALLS   #2, CRD$PRINT_DS_COMMAND         ;
        50 00500001   8F  D0 0004C     MOVL    #5242881, R0           ; 0837
     04 00053     RET                 ; 0838

; Routine Size:  84 bytes,    Routine Base:  CODE + 0374
```

```
;    0839  1 %SBTTL 'CRD$Load_Select_Com'
;    0840  1
;    0841  1 GLOBAL ROUTINE CRD$Load_Select_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0842  1
;    0843  1 !++
;    0844  1 ! Functional Description:
;    0845  1 !
;    0846  1 ! Load the next SELECT command into the CLI buffer.
;    0847  1 !
;    0848  1 ! Formal Parameters:
;    0849  1 !
;    0850  1 ! CLI_Buffer_Length  : The length of the CLI buffer.
;    0851  1 ! CLI_Buffer_Address : The address of the CLI buffer.
;    0852  1 !
;    0853  1 ! Side Effects:
;    0854  1 !
;    0855  1 ! None
;    0856  1 !
;    0857  1 ! Completion Codes:
;    0858  1 !
;    0859  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0860  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0861  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0862  1 !--
```

```
;    0863  2 BEGIN      ! Routine CRD$Load_Select_Com
;    0864  2  MAP
;    0865  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0866  2
;    0867  2  MAP
;    0868  2   CRD$GT_DS_SELECT_Com  : VECTOR [, BYTE];
;    0869  2
;    0870  2  BIND
;    0871  3   Diag_Vector_Ptr = (.CRD$GA_Current_Diagnostic [CRD$K_DDB_Auto_Support_Entry])
;    0872  2    : Hardware_Support_Vector;
;    0873  2
;    0874  2  BIND
;    0875  2   Device_Name = (.Diag_Vector_Ptr [CRD$K_Device_Name]) : VECTOR [, BYTE];
;    0876  2
;    0877  2  IF (.Device_Name [0] EQL 0) THEN
;    0878  3   BEGIN
;    0879  3   !+
;    0880  3   ! Nothing to Select, so do nothing.
;    0881  3   !-
;    0882  4   RETURN (CRD$K_Repeat_Turing)
;    0883  3   END              ![13]
;    0884  2  ELSE               ![13]
;    0885  2   !+
;    0886  2   ! DS> Select <device_name>
;    0887  2   !-
;    0888  2   CH$COPY (
;    0889  2    .CRD$GT_DS_SELECT_Com [0], CH$PTR (CRD$GT_DS_SELECT_Com [1]),
;    0890  2    .Device_Name [0],     CH$PTR (Device_Name [1]),
;    0891  2    %C' ',
;    0892  2    .CLI_Buffer_Length,    CH$PTR (.CLI_Buffer_Address)
;    0893  2    );
;    0894  2
;    0895  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0896  2
;    0897  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0898  1 END;      ! Routine CRD$Load_Select_Com


        .EXTRN  CRD$GT_DS_SELECT_COM

          07FC 00000     .ENTRY  CRD$LOAD_SELECT_COM, Save R2,R3,R4,R5,R6,-   ; 0841
         R7,R8,R9,R10            ;
        50 00000000G   EF  D0 00002    MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0871
        50 0C    A0  D0 00009    MOVL    12(R0), R0           ;
        57 0C    A0  D0 0000D    MOVL    12(R0), R7           ; 0875
        67  95 00011    TSTB    (R7)              ; 0877
        04  12 00013    BNEQ    1$              ;
        50       02  D0 00015    MOVL    #2, R0           ; 0882
    04 00018    RET                 ;
        5A 00000000G   EF  9A 00019 1$:   MOVZBL  CRD$GT_DS_SELECT_COM, R10        ; 0889
        59       67  9A 00020    MOVZBL  (R7), R9           ; 0890
        58 04    AC  D0 00023    MOVL    CLI_BUFFER_LENGTH, R8         ; 0892
        56 08    AC  D0 00027    MOVL    CLI_BUFFER_ADDRESS, R6        ;
58          20 00000000G   EF      5A  2C 0002B    MOVC5   R10, CRD$GT_DS_SELECT_COM+1, #32, R8, (R6)  ;
```

```
            66     00034                ;
            0D  18 00035     BGEQ    2$                ;
            56      5A  C0 00037     ADDL2   R10, R6           ;
            58      5A  C2 0003A     SUBL2   R10, R8           ;
    58           20   01   A7      59  2C 0003D    MOVC5   R9, 1(R7), #32, R8, (R6)        ;
            66     00043              ;
            7E 04   AC  7D 00044 2$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)         ; 0895
       00000000G  EF      02  FB 00048    CALLS   #2, CRD$PRINT_DS_COMMAND          ;
            50 00500001   8F  D0 0004F    MOVL    #5242881, R0           ; 0897
         04 00056    RET                 ; 0898

; Routine Size:  87 bytes,    Routine Base:  CODE + 03C8
```

```
;    0899  1 %SBTTL 'CRD$Load_Set_Event_Com'
;    0900  1
;    0901  1 GLOBAL ROUTINE CRD$Load_Set_Event_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0902  1
;    0903  1 !++
;    0904  1 ! Functional Description:
;    0905  1 !
;    0906  1 ! Load the next SET EVENT command into the CLI buffer.
;    0907  1 !
;    0908  1 ! Formal Parameters:
;    0909  1 !
;    0910  1 ! CLI_Buffer_Length  : The length of the CLI buffer.
;    0911  1 ! CLI_Buffer_Address : The address of the CLI buffer.
;    0912  1 !
;    0913  1 ! Side Effects:
;    0914  1 !
;    0915  1 ! None
;    0916  1 !
;    0917  1 ! Completion Codes:
;    0918  1 !
;    0919  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0920  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0921  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0922  1 !--
```

```
;    0923  2 BEGIN     ! Routine CRD$Load_Set_Event_Com
;    0924  2  MAP
;    0925  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0926  2
;    0927  2  MAP
;    0928  2   CRD$GT_DS_SET_EVENT_Com : VECTOR [, BYTE];
;    0929  2
;    0930  2  BIND
;    0931  3   Diag_Vector_Ptr = (.CRD$GA_Current_Diagnostic [CRD$K_DDB_Auto_Support_Entry])
;    0932  2    : Hardware_Support_Vector;
;    0933  2
;    0934  2  BIND
;    0935  2   Set_Event_Args = (.Diag_Vector_Ptr [CRD$K_Set_Event_Args]) : VECTOR [, BYTE];
;    0936  2
;    0937  2  IF (.Set_Event_Args [0] EQL 0) THEN
;    0938  3   RETURN (CRD$K_Repeat_Turing)
;    0939  2  ELSE
;    0940  2   CH$COPY (
;    0941  2    .CRD$GT_DS_SET_EVENT_Com [0], CH$PTR (CRD$GT_DS_SET_EVENT_Com [1]),
;    0942  2    .Set_Event_Args [0],       CH$PTR (Set_Event_Args [1]),
;    0943  2    %C' ',
;    0944  2    .CLI_Buffer_Length,       CH$PTR (.CLI_Buffer_Address)
;    0945  2    );
;    0946  2
;    0947  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0948  2
;    0949  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0950  1 END;     ! Routine CRD$Load_Set_Event_Com


        .EXTRN  CRD$GT_DS_SET_EVENT_COM

          07FC 00000     .ENTRY  CRD$LOAD_SET_EVENT_COM, Save R2,R3,R4,R5,-  ; 0901
         R6,R7,R8,R9,R10          ;
        50 00000000G  EF  D0 00002     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0931
        50 0C    A0  D0 00009     MOVL    12(R0), R0           ;
        57 08    A0  D0 0000D     MOVL    8(R0), R7          ; 0935
        67  95 00011     TSTB    (R7)             ; 0937
        04  12 00013     BNEQ    1$             ;
        50       02  D0 00015     MOVL    #2, R0           ; 0938
     04 00018     RET                    ;
        5A 00000000G  EF  9A 00019 1$:   MOVZBL  CRD$GT_DS_SET_EVENT_COM, R10        ; 0941
        59       67  9A 00020     MOVZBL  (R7), R9           ; 0942
        58 04    AC  D0 00023     MOVL    CLI_BUFFER_LENGTH, R8          ; 0944
        56 08    AC  D0 00027     MOVL    CLI_BUFFER_ADDRESS, R6         ;
    58          20 00000000G  EF       5A  2C 0002B    MOVC5   R10, CRD$GT_DS_SET_EVENT_COM+1, #32, R8, -  ;
        66       00034     (R6)           ;
        0D  18 00035     BGEQ    2$             ;
        56       5A  C0 00037     ADDL2   R10, R6          ;
        58       5A  C2 0003A     SUBL2   R10, R8          ;
    58          20    01    A7       59  2C 0003D    MOVC5   R9, 1(R7), #32, R8, (R6)        ;
        66       00043          ;
        7E 04    AC  7D 00044 2$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)        ; 0947
     00000000G  EF       02  FB 00048     CALLS   #2, CRD$PRINT_DS_COMMAND        ;
```

```
           50 00500001   8F  D0 0004F     MOVL    #5242881, R0          ; 0949
        04 00056    RET                 ; 0950

; Routine Size:  87 bytes,    Routine Base:  CODE + 041F
```

```
;    0951  1 %SBTTL 'CRD$Load_Set_Flags_Com'
;    0952  1
;    0953  1 GLOBAL ROUTINE CRD$Load_Set_Flags_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0954  1
;    0955  1 !++
;    0956  1 ! Functional Description:
;    0957  1 !
;    0958  1 ! Load the next SET FLAGS command into the CLI buffer.
;    0959  1 !
;    0960  1 ! Formal Parameters:
;    0961  1 !
;    0962  1 ! CLI_Buffer_Length  : The length of the CLI buffer.
;    0963  1 ! CLI_Buffer_Address : The address of the CLI buffer.
;    0964  1 !
;    0965  1 ! Side Effects:
;    0966  1 !
;    0967  1 ! None
;    0968  1 !
;    0969  1 ! Completion Codes:
;    0970  1 !
;    0971  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    0972  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    0973  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    0974  1 !--
```

```
;    0975  2 BEGIN     ! Routine CRD$Load_Set_Flags_Com
;    0976  2  MAP
;    0977  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    0978  2
;    0979  2  MAP
;    0980  2   CRD$GT_DS_SET_FLAGS_Com : VECTOR [, BYTE];
;    0981  2
;    0982  2  BIND
;    0983  3   Diag_Vector_Ptr = (.CRD$GA_Current_Diagnostic [CRD$K_DDB_Auto_Support_Entry])
;    0984  2     : Hardware_Support_Vector;
;    0985  2
;    0986  2  BIND
;    0987  2   Set_Flags_Args = (.Diag_Vector_Ptr [CRD$K_Set_Flags_Args]) : VECTOR [, BYTE];
;    0988  2
;    0989  2  IF (.Set_Flags_Args [0] EQL 0) THEN
;    0990  3   RETURN (CRD$K_Repeat_Turing)
;    0991  2  ELSE
;    0992  2   CH$COPY (
;    0993  2     .CRD$GT_DS_SET_FLAGS_Com [0], CH$PTR (CRD$GT_DS_SET_FLAGS_Com [1]),
;    0994  2     .Set_Flags_Args [0],       CH$PTR (Set_Flags_Args [1]),
;    0995  2     %C' ',
;    0996  2     .CLI_Buffer_Length,       CH$PTR (.CLI_Buffer_Address)
;    0997  2     );
;    0998  2
;    0999  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    1000  2
;    1001  2  $CRD_Return_Length_Status (CRD$K_Success);
;    1002  1 END;     ! Routine CRD$Load_Set_Flags_Com
```

```
          07FC 00000     .ENTRY  CRD$LOAD_SET_FLAGS_COM, Save R2,R3,R4,R5,-  ; 0953
         R6,R7,R8,R9,R10          ;
        50 00000000G  EF  D0 00002     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 0983
        50 0C    A0  D0 00009     MOVL    12(R0), R0          ;
        57 04    A0  D0 0000D     MOVL    4(R0), R7          ; 0987
        67  95 00011     TSTB    (R7)              ; 0989
        04  12 00013     BNEQ    1$              ;
        50       02  D0 00015     MOVL    #2, R0           ; 0990
     04 00018     RET              ;
        5A 00000000G  EF  9A 00019 1$:   MOVZBL  CRD$GT_DS_SET_FLAGS_COM, R10        ; 0993
        59       67  9A 00020     MOVZBL  (R7), R9          ; 0994
        58 04    AC  D0 00023     MOVL    CLI_BUFFER_LENGTH, R8          ; 0996
        56 08    AC  D0 00027     MOVL    CLI_BUFFER_ADDRESS, R6          ;
58           20 00000000G  EF       5A  2C 0002B     MOVC5   R10, CRD$GT_DS_SET_FLAGS_COM+1, #32, R8,     ;
        66     00034     (R6)             ;
        0D  18 00035     BGEQ    2$              ;
        56       5A  C0 00037     ADDL2   R10, R6           ;
        58       5A  C2 0003A     SUBL2   R10, R8           ;
58           20    01    A7       59  2C 0003D     MOVC5   R9, 1(R7), #32, R8, (R6)         ;
        66     00043          ;
        7E 04    AC  7D 00044 2$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)         ; 0999
    00000000G  EF       02  FB 00048     CALLS   #2, CRD$PRINT_DS_COMMAND         ;
```

```
          50 00500001   8F  D0 0004F    MOVL    #5242881, R0          ; 1001
       04 00056    RET                 ; 1002

; Routine Size:  87 bytes,    Routine Base:  CODE + 0476
```

```
;    1003  1 %SBTTL 'CRD$Load_Start_Com'
;    1004  1
;    1005  1 GLOBAL ROUTINE CRD$Load_Start_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    1006  1
;    1007  1 !++
;    1008  1 ! Functional Description:
;    1009  1 !
;    1010  1 ! Load the next START command into the CLI buffer.
;    1011  1 !
;    1012  1 ! Formal Parameters:
;    1013  1 !
;    1014  1 ! CLI_Buffer_Length  : The length of the CLI buffer.
;    1015  1 ! CLI_Buffer_Address : The address of the CLI buffer.
;    1016  1 !
;    1017  1 ! Side Effects:
;    1018  1 !
;    1019  1 ! None
;    1020  1 !
;    1021  1 ! Completion Codes:
;    1022  1 !
;    1023  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    1024  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    1025  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    1026  1 !--
```

```
;    1027  2 BEGIN     ! Routine CRD$Load_Start_Com
;    1028  2  MAP
;    1029  2   CRD$GT_DS_START_Com : VECTOR [, BYTE];
;    1030  2
;    1031  2  MAP
;    1032  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    1033  2
;    1034  2  BIND
;    1035  3   Diag_Vector_Ptr = (.CRD$GA_Current_Diagnostic [CRD$K_DDB_Auto_Support_Entry])
;    1036  2    : Hardware_Support_Vector;
;    1037  2
;    1038  2  BIND
;    1039  2   Start_Qualifiers = (.Diag_Vector_Ptr [CRD$K_Start_Qualifiers]) : VECTOR [, BYTE];
;    1040  2
;    1041  2  CH$COPY (
;    1042  2   .CRD$GT_DS_START_Com [0], CH$PTR (CRD$GT_DS_START_Com [1]),
;    1043  2   .Start_Qualifiers [0],   CH$PTR (Start_Qualifiers [1]),
;    1044  2   %C' ',
;    1045  2   .CLI_Buffer_Length,   CH$PTR (.CLI_Buffer_Address)
;    1046  2   );
;    1047  2
;    1048  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    1049  2
;    1050  2  $CRD_Return_Length_Status (CRD$K_Success);
;    1051  1 END;     ! Routine CRD$Load_Start_Com


          .EXTRN  CRD$GT_DS_START_COM

            07FC 00000     .ENTRY  CRD$LOAD_START_COM, Save R2,R3,R4,R5,R6,R7,-; 1005
           R8,R9,R10           ;
          50 00000000G  EF  D0 00002     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0        ; 1035
          50 0C    A0  D0 00009     MOVL    12(R0), R0           ;
          58 10    A0  D0 0000D     MOVL    16(R0), R8           ; 1039
          5A 00000000G  EF  9A 00011     MOVZBL  CRD$GT_DS_START_COM, R10        ; 1042
          59       68  9A 00018     MOVZBL  (R8), R9           ; 1043
          57 04    AC  D0 0001B     MOVL    CLI_BUFFER_LENGTH, R7          ; 1045
          56 08    AC  D0 0001F     MOVL    CLI_BUFFER_ADDRESS, R6         ;
    57          20 00000000G  EF       5A  2C 00023     MOVC5   R10, CRD$GT_DS_START_COM+1, #32, R7, (R6)    ;
          66       0002C           ;
          0D  18 0002D     BGEQ    1$           ;
          56       5A  C0 0002F     ADDL2   R10, R6          ;
          57       5A  C2 00032     SUBL2   R10, R7          ;
    57          20   01    A8       59  2C 00035     MOVC5   R9, 1(R8), #32, R7, (R6)        ;
          66       0003B           ;
          7E 04    AC  7D 0003C 1$:    MOVQ    CLI_BUFFER_LENGTH, -(SP)        ; 1048
       00000000G  EF       02  FB 00040     CALLS   #2, CRD$PRINT_DS_COMMAND        ;
          50 00500001   8F  D0 00047     MOVL    #5242881, R0           ; 1050
           04 0004E     RET                 ; 1051

; Routine Size:  79 bytes,    Routine Base:  CODE + 04CD
```

```
;    1052  1 %SBTTL 'CRD$Set_Up_Diagnostic'
;    1053  1
;    1054  1 GLOBAL ROUTINE CRD$Set_Up_Diagnostic =          ![16]
;    1055  1 !++                 ! |
;    1056  1 ! Functional Description:          ! V
;    1057  1 !
;    1058  1 ! Print the testing started message, check to see if the device was attached; if not,
;    1059  1 ! print an error message and go to advance diagnostic; if so, then repeat_turing.
;    1060  1 !
;    1061  1 ! Formal Parameters:
;    1062  1 !
;    1063  1 ! None
;    1064  1 !
;    1065  1 ! Side Effects:
;    1066  1 !
;    1067  1 ! None
;    1068  1 !
;    1069  1 ! Completion Codes:
;    1070  1 !
;    1071  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    1072  1 ! CRD$K_Success => Return success to the CRD$Turing_Machine routine.
;    1073  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    1074  1 !--
```

```
;    1075  2 BEGIN     ! Routine CRD$Set_Up_Diagnostic
;    1076  2  MAP
;    1077  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    1078  2
;    1079  2  BIND
;    1080  3   Diag_Vector_Ptr = (.CRD$GA_Current_Diagnostic [CRD$K_DDB_Auto Support_Entry])
;    1081  2    : Hardware_Support_Vector;
;    1082  2
;    1083  2  IF (.CRD$GB_CRD_Debug) THEN
;    1084  2   $CRD_Print ($ASCIC ('*** Set_Up_Diagnostic:!/'));
;    1085  2
;    1086  2  CRD$Print_DDB (.CRD$GA_Current_Diagnostic);
;    1087  2
;    1088  2  !+
;    1089  2  ! Now output the "TESTING STARTED" message. This must be done here because this message must be
;    1090  2  ! output regardless of whether or not the diagnostic is run.
;    1091  2  !
;    1092  2  ! Testing_Started : Diag_Vector_Ptr [CRD$K_Test_Start_Text]
;    1093  2  ! RunTime    : Diag_Vector_Ptr [CRD$K_RunTime]
;    1094  2  !
;    1095  2  ! If the DDB_Status_Check_Dev_Type bit is True, then the device type must be extracted from the
;    1096  2  ! PTable and printed instead of the device-type field in the Test_Start_Text.
;    1097  2  !-
;    1098  2
; P  1099  2  $CRD_Concat (CRD$GA_CRD_Message_Buffer, CRD$K_CRD_Message_Buffer_Len,
;    1100  2   CRD$GT_Testing_Device, CRD$GT_Testing_Started, CRD$GT_RunTime);
;    1101  2      ! Concatenate the messages (i.e., Ascic strings) so that only one
;    1102  2      ! print statement is needed to output the testing started message.
;    1103  2
;    1104  2
;    1105  3  IF (
;    1106  4   (.CRD$GA_Current_Diagnostic [CRD$K_DDB_PTable_Entry] NEQ 0)
;    1107  3  AND
;    1108  4   (.CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Check_Device_Type>)
;    1109  2  ) THEN
;    1110  3   BEGIN
;    1111  3   !+
;    1112  3   ! There is a PTable for this device, and we must replace the device type field in the
;    1113  3   ! Test_Start_Text with the device type that is in the Ptable. This is mainly because the
;    1114  3   ! 11/730,725 packaged systems may come with either an RL02 as a diagnostic disk or with an
;    1115  3   ! R80 diagnostic disk. Because it can be either one, I have to print the correct one when
;    1116  3   ! I print the Testing_Started message.
;    1117  3   !-
;    1118  3
;    1119  3   LOCAL
;    1120  3    PTable_Device_Type_Length;
;    1121  3      ! The length of the Device_Type string in the PTable.
;    1122  3
;    1123  3   LOCAL
;    1124  3    Temp_Test_Start_Text : VECTOR [CRD$K_CRD_Message_Buffer_Len, BYTE];
;    1125  3      ! Temporarily store the Test_Start_Text from the Hardware_Support_Table in
;    1126  3      ! here so that it can be modified (it cannot be modified directly in the
;    1127  3      ! Table because the Table is in a write-protected PSect.
;    1128  3
;    1129  3   BIND
```

```
;    1130  3     Ptable_Ptr = (.CRD$GA_Current_Diagnostic [CRD$K_DDB_PTable_Entry]);
;    1131  3
;    1132  3    BIND
;    1133  3     Ptable = (PTable_Ptr) : $DS_HPO_DECL ();
;    1134  3
;    1135  3    BIND
;    1136  3     PTable_Device_Type = (PTable [HP$T_Type]) : VECTOR [, BYTE];
;    1137  3       ! This is the ASCIC Device Type string in the ptable
;    1138  3
;    1139  3    BIND
;    1140  3     Test_Start_Text = (.Diag_Vector_Ptr [CRD$K_Test_Start_Text]) : VECTOR [, BYTE];
;    1141  3       ! Reference the actual Test_Start_Text as a vector of bytes.
;    1142  3
;    1143  3    BUILTIN
;    1144  3     MOVC3;
;    1145  3
;    1146  3    MOVC3 (%REF (.Test_Start_Text [0] + 1), Test_Start_Text, Temp_Test_Start_Text);
;    1147  3       ! Move the actual Test_Start_Text Ascic string to the Temp_Test_Start_Text
;    1148  3       ! buffer (i.e., the whole string - e.g., 'R80         DQA0 '). Use %REF
;    1149  3       ! because MovC3 requires an address. Use '+ 1  so that the count byte itself
;    1150  3       ! is included in the transfer.
;    1151  3
;    1152  3    PTable_Device_Type_Length = .PTable_Device_Type [0];
;    1153  3       ! Get the length of the PTable's device type string (i.e., 4 for 'RL02').
;    1154  3
;    1155  3    MOVC3 (PTable_Device_Type_Length, PTable_Device_Type [1], Temp_Test_Start_Text [1]);
;    1156  3       ! Now overwrite whatever was there in the Test_Start_Text with the new device
;    1157  3       ! type string from the PTable. Now we can use Temp_Test_Start_Text as the new
;    1158  3       ! actual Test_Start_Text. Using the example above, overwrite the 'R80 ' with
;    1159  3       ! the 'RL02' from the PTable.
;    1160  3
; P  1161  3     $CRD_Print (CRD$GA_CRD_Message_Buffer,
; P  1162  3      Temp_Test_Start_Text,
;    1163  4      .Diag_Vector_Ptr [CRD$K_RunTime])
;    1164  4        ! Use the Temp_Test_Start_Text buffer instead of the actual Test_Start_Text,
;    1165  4        ! as was used below.
;    1166  3     END
;    1167  2    ELSE
;    1168  3     BEGIN
; P  1169  3     $CRD_Print (CRD$GA_CRD_Message_Buffer,
; P  1170  3      .Diag_Vector_Ptr [CRD$K_Test_Start_Text],
;    1171  4      .Diag_Vector_Ptr [CRD$K_RunTime])
;    1172  4        ! Use the actual Test_Start_Text string
;    1173  2     END;
;    1174  2
;    1175  2    !+
;    1176  2    ! Now we must check the diagnostic to see if there is a PTable present for this device. If not,
;    1177  2    ! print a message saying that testing cannot be done on this device. If there is a Ptable, carry on.
;    1178  2    !-
;    1179  2
;    1180  2    IF (.CRD$GA_Current_Diagnostic [CRD$K_DDB_PTable_Entry] EQL 0) THEN
;    1181  3     BEGIN
;    1182  3     $CRD_Print (CRD$GT_Dev_Unavailable);
;    1183  3       ! Print a message about the device being unavailable for testing
;    1184  3
```

```
;   1185  3   CRD$GL_Current_State = CRD$K_State_Advance_Diagnostic;
;   1186  3     ! Skip to the next diagnostic in the list of DDBs.
;   1187  3
;   1188  3   CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed>   - True;
;   1189  3     ! We did print a message
;   1190  3
;   1191  3   CRD$GA_Current_Diagnostic [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete>   False;
;   1192  3     ! Testing wasn't completed (wasn't even started) on this device.
;   1193  2   END;
;   1194  2                 ! ^
;   1195  2  IF (.CRD$GB_CRD_Debug) THEN
;   1196  2   $CRD_Print ($ASCIC ('Set_Up_Diagnostic: at the end:!/'));
;   1197  2
;   1198  2  CRD$Print_DDB (.CRD$GA_Current_Diagnostic);
;   1199  2
;   1200  2  RETURN (CRD$K_Repeat_Turing);          ! |
;   1201  1 END;    ! Routine CRD$Set_Up_Diagnostic       ![16]



          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

61  69  44  5F  70  55  5F  74  65  53  20  2A  2A  2A  18  000EA P.AAM:  .ASCII  <24>\*** Set_Up_Diagnostic:!/\       ;
      2F  21  3A  63  69  74  73  6F  6E  67  000F9                          ;
73  6F  6E  67  61  69  44  5F  70  55  5F  74  65  53  20  00103 P.AAN:  .ASCII  \ Set_Up_Diagnostic: at the end:!/\       ;
64  6E  65  20  65  68  74  20  74  61  20  3A  63  69  74  00112              ;
      2F  21  3A  00121               ;

      LENGTH=       132
       .EXTRN  CRD$GA_CRD_MESSAGE_BUFFER
       .EXTRN  CRD$GT_TESTING_DEVICE
       .EXTRN  CRD$GT_TESTING_STARTED
       .EXTRN  CRD$GT_RUNTIME, CRD$GT_DEV_UNAVAILABLE

       .PSECT  CODE,NOWRT,  SHR,  PIC,2

         0FFC 00000     .ENTRY  CRD$SET_UP_DIAGNOSTIC, Save R2,R3,R4,R5,R6,-; 1054
        R7,R8,R9,R10,R11         ;
        5E      FF7C    CE  9E 00002    MOVAB   -132(SP), SP            ;
        50 00000000G     F  D0 00007    MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0          ; 1080
        56 0C    A0  Du 0000E    MOVL    12(R0), R6              ;
        11 00000000G  EF  E9 00012    BLBC    CRD$GB_CRD_DEBUG, 1$           ; 1083
    00000000'  EF  9F 00019    PUSHAB  P.AAM             ; 1084
        01  DD 0001F    PUSHL   #1              ;
        1A  DD 00021    PUSHL   #26             ;
    00000000G  FF       03  FB 00023    CALLS   #3, @CRD$PRINT          ;
    00000000G  EF  DD 0002A 1$:   PUSHL   CRD$GA_CURRENT_DIAGNOSTIC         ; 1086
    00000000G  EF       01  FB 00030    CALLS   #1, CRD$PRINT_DDB           ;
0084    8F          20          6E      00  2C 00037    MOVC5   #0, (SP), #32, #132, ADDRESS        ; 1100
    00000000G  EF      0003E                ;
        50 00000000G  EF  9A 00043    MOVZBL  STRING, R0              ;
        51 00000000G  EF  9A 0004A    MOVZBL  STRING, R1              ;
        50      51  C0 00051    ADDL2   R1, R0              ;
    00000000G  EF         50 00000000G  EF  81 00054    ADDB3   STRING, R0, ADDRESS         ;
        5B 00000000G  EF  9A 00060    MOVZBL  STRING, R11             ;
```

```
                5A 00000000G  EF  9A 00067     MOVZBL  STRING, R10              ;
                59 00000000G  EF  9A 0006E     MOVZBL  STRING, R9               ;
                58 83   8F  9A 00075     MOVZBL  #131, R8                 ;
                57 00000000G  EF  9E 00079     MOVAB   ADDRESS+1, R7                ;
    58           20 00000000G  EF      5B  2C 00080     MOVC5   R11, CRD$GT_TESTING_DEVICE+1, #32, R8, (R7) ;
                67     00089                ;
                22  18 0008A     BGEQ    2$                ;
                57      5B  C0 0008C     ADDL2   R11, R7           ;
                58      5B  C2 0008F     SUBL2   R11, R8           ;
    58           20 00000000G  EF      5A  2C 00092     MOVC5   R10, CRD$GT_TESTING_STARTED+1, #32, R8, -    ;
                67     0009B    (R7)            ;
                10  18 0009C     BGEQ    2$                ;
                57      5A  C0 0009E     ADDL2   R10, R7           ;
                58      5A  C2 000A1     SUBL2   R10, R8           ;
    58           20 00000000G  EF      59  2C 000A4     MOVC5   R9, CRD$GT_RUNTIME+1, #32, R8, (R7)        ;
                67     000AD                ;
                50 00000000G  EF  D0 000AE 2$:   MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0         ; 1106
      08        A0  D5 000B5     TSTL    8(R0)             ;
                29  13 000B8     BEQL    3$                ;
                24   14   A0       03  E1 000BA     BBC     #3, 20(R0), 3$           ; 1108
                50 08   A0  D0 000BF     MOVL    8(R0), R0             ; 1136
                57 26   A0  9E 000C3     MOVAB   38(R0), R7             ;
                50 14   B6  9A 000C7     MOVZBL  @20(R6), R0            ; 1146
                50  D6 000CB     INCL    R0                ;
                6E   14   B6       50  28 000CD     MOVC3   R0, @20(R6), TEMP_TEST_START_TEXT        ;
                50       67  9A 000D2     MOVZBL  (R7), PTABLE_DEVICE_TYPE_LENGTH        ; 1152
      01        AE   01   A7       50  28 000D5     MOVC3   PTABLE_DEVICE_TYPE_LENGTH, 1(R7), -        ; 1155
                 TEMP_TEST_START_TEXT+1          ;
      18        A6  DD 000DB     PUSHL   24(R6)            ; 1163
      04        AE  9F 000DE     PUSHAB  TEMP_TEST_START_TEXT          ;
                04  11 000E1     BRB     4$                ;
                7E 14   A6  7D 000E3 3$:   MOVQ    20(R6), -(SP)           ; 1171
       00000000G  EF  9F 000E7 4$:   PUSHAB  CRD$GA_CRD_MESSAGE_BUFFER         ;
                01  DD 000ED     PUSHL   #1                ;
                1A  DD 000EF     PUSHL   #26               ;
       00000000G  FF       05  FB 000F1     CALLS   #5, @CRD$PRINT           ;
                50 00000000G  EF  D0 000F8     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0         ; 1180
      08        A0  D5 000FF     TSTL    8(R0)              ;
                28  12 00102     BNEQ    5$                ;
       00000000G  EF  9F 00104     PUSHAB  CRD$GT_DEV_UNAVAILABLE         ; 1182
                01  DD 0010A     PUSHL   #1                ;
                1A  DD 0010C     PUSHL   #26               ;
       00000000G  FF       03  FB 0010E     CALLS   #3, @CRD$PRINT           ;
       00000000G  EF       19  D0 00115     MOVL    #25, CRD$GL_CURRENT_STATE         ; 1185
                50 00000000G  EF  D0 0011C     MOVL    CRD$GA_CURRENT_DIAGNOSTIC, R0         ; 1188
      14        A0 40   8F  88 00123     BISB2   #64, 20(R0)            ;
      14        A0       20  8A 00128     BICB2   #32, 20(R0)            ; 1191
                11 00000000G  EF  E9 0012C 5$:   BLBC    CRD$GB_CRD_DEBUG, 6$          ; 1195
       00000000'  EF  9F 00133     PUSHAB  P.AAN             ; 1196
                01  DD 00139     PUSHL   #1                ;
                1A  DD 0013B     PUSHL   #26               ;
       00000000G  FF       03  FB 0013D     CALLS   #3, @CRD$PRINT           ;
       00000000G  EF  DD 00144 6$:   PUSHL   CRD$GA_CURRENT_DIAGNOSTIC         ; 1198
       00000000G  EF       01  FB 0014A     CALLS   #1, CRD$PRINT_DDB          ;
                50       02  D0 00151     MOVL    #2, R0            ; 1200
```

```
        04 00154    RET                 ; 1201

; Routine Size:  341 bytes,    Routine Base:  CODE + 051C
```

```
;    1202  1 %SBTTL 'CRD$Start_AutoSizer'
;    1203  1
;    1204  1 GLOBAL ROUTINE CRD$Start_AutoSizer (CLI_Buffer_Length, CLI_Buffer_Address) =
;    1205  1
;    1206  1 !++
;    1207  1 ! Functional Description:
;    1208  1 !
;    1209  1 ! Load the START command for the AutoSizer.
;    1210  1 !
;    1211  1 ! Formal Parameters:
;    1212  1 !
;    1213  1 ! CLI_Buffer_Length  : The length of the CLI buffer.
;    1214  1 ! CLI_Buffer_Address : The address of the CLI buffer.
;    1215  1 !
;    1216  1 ! Side Effects:
;    1217  1 !
;    1218  1 ! None
;    1219  1 !
;    1220  1 ! Completion Codes:
;    1221  1 !
;    1222  1 ! CRD$K_Failure => Return failure to the CRD$Turing_Machine routine.
;    1223  1 ! CRD$K_Success -> Return success to the CRD$Turing_Machine routine.
;    1224  1 ! CRD$K_Repeat_Turing  => Cause the Turing_Machine routine to call itself recursively.
;    1225  1 !--
```

```
;    1226  2 BEGIN     ! Routine CRD$Start_AutoSizer
;    1227  2  MAP
;    1228  2   CRD$GA_Current_Diagnostic : REF Device_Data_Block_Structure;
;    1229  2
;    1230  2  MAP
;    1231  2   CRD$GT_DS_START_Com : VECTOR [, BYTE];
;    1232  2
;    1233  2  CH$COPY (              ![13]
;    1234  2   .CRD$GT_DS_START_Com [0], CH$PTR (CRD$GT_DS_START_Com [1]),     ![13]
;    1235  2   %C' ',                ![13]
;    1236  2   .CLI_Buffer_Length,   CH$PTR (.CLI_Buffer_Address)
;    1237  2   );
;    1238  2
;    1239  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    1240  2
;    1241  2  $CRD_Return_Length_Status (CRD$K_Success);
;    1242  1 END;     ! Routine CRD$Start_AutoSizer




              003C 00000     .ENTRY  CRD$START_AUTOSIZER, Save R2,R3,R4,R5       ; 1204
           50 00000000G   EF   9A 00002     MOVZBL  CRD$GT_DS_START_COM, R0        ; 1234
      04   AC           20 00000000G   EF      50   2C 00009     MOVC5   R0, CRD$GT_DS_START_COM+1, #32, -       ; 1236
      08   BC      00013     CLI_BUFFER_LENGTH, @CLI_BUFFER_ADDRESS      ;
           7E 04    AC   7D 00015     MOVQ    CLI_BUFFER_LENGTH, -(SP)        ; 1239
    00000000G   EF       02   FB 00019     CALLS   #2, CRD$PRINT_DS_COMMAND        ;
           50 00500001    8F   D0 00020     MOVL    #5242881, R0           ; 1241
            04 00027     RET                 ; 1242

; Routine Size:  40 bytes,    Routine Base:  CODE + 0671
```

```
:   1243  1 END     ! End of CRDACT2 module
:   1244  0 ELUDOM
```

```
:            PSECT SUMMARY
:
: Name    Bytes          Attributes
:
:  DATA          292  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
:  CODE         1689  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$ASCIC     Macro    Lib03   265     277     318     517     518     519     521     523     525     664
              768     1084    1196
$CRD_CAT    Macro    Lib01  1100
$CRD_CONCAT    Macro    Lib01  1099
$CRD_LENGTH    Macro    Lib01  1100
$CRD_LITERALS    Macro    Lib01   211
$CRD_PRINT    Macro    Lib01   262     265     277     318     664     693     1084    1161    1169    1182
              1196
$CRD_RETURN_LENGTH_STATUS Macro    Lib01   462     737     784     837     897     949     1001    1050    1241
$DS_HPO_DECL    Macro    Lib03   538     1133
$DS_HPO_F    Fieldset Lib03   538     1133
$DS_TYPEDEF    Macro    Lib03   210     262     265     277     319     665     693     1084    1163    1171
              1182    1196
$EQULST     Macro    Lib04   210     211     262     265     277     319     665     693     1084    1163
              1171    1182    1196
$ER     Comptime   227
$MODULE     Bind       227
ADDRESS     Bind      1100  1100=   1100m
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     211
AUTO$K_EXIT_CONTROL_C  Literal     211
AUTO$K_EXIT_DUMMY_2  Literal     211
AUTO$K_EXIT_DUMMY_3  Literal     211
AUTO$K_EXIT_ERRORS_COMPLETE Literal     211
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     211
AUTO$K_EXIT_INTERNAL_ERROR Literal     211
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     211   375
AUTO$K_EXIT_LAST_REASON  Literal     211   211
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     211
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     211
AUTO$K_EXIT_NOERRORS_PREP Literal     211
CLI_BUFFER_ADDRESS  RoutForm   416   457a=   460.
    RoutForm   699   732a=   735.
    RoutForm   790   832a=   835.
    RoutForm   841   892a=   895.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    RoutForm   901   944a=   947.
    RoutForm   953   996a=   999.
    RoutForm  1005  1045a=  1048.
    RoutForm  1204  1236a=  1239.
CLI_BUFFER_LENGTH  RoutForm   416   457.    460.
    RoutForm   699   732.    735.
    RoutForm   790   832.    835.
    RoutForm   841   892.    895.
    RoutForm   901   944.    947.
    RoutForm   953   996.    999.
    RoutForm  1005  1045.   1048.
    RoutForm  1204  1236.   1239.
COMMAND_OUTPUT   Bind       768
COMMAND_PTR   Local      765   776=    779a.
CRD$ADVANCE_DIAGNOSTIC  GlobRout   230 Lib01e    160f
CRD$ADVANCE_GENERAL_COM  GlobRout   293 Lib01e    163f
CRD$AL_IDC_HDWR_SPRT_TABLE External Lib01   555a
CRD$AL_LESI_HDWR_SPRT_TABLE External Lib01   568a
CRD$AL_UDA_0_HDWR_SPRT_TABLE External Lib01   612a    632a
CRD$AL_UDA_1_HDWR_SPRT_TABLE External Lib01   601a
CRD$AL_UDA_2_HDWR_SPRT_TABLE External Lib01   623a
CRD$AL_UDA_3_HDWR_SPRT_TABLE External Lib01   589a
CRD$ASSIGN_PTABLE_PTRS  GlobRout   341 Lib01e    166f
CRD$AUTOSIZER_SET_FLAGS  GlobRout   416 Lib01e    169f
CRD$BUILD_DDBS   ExtRout  Lib01   380c
CRD$DETERMINE_BASE_SYSTEM GlobRout   470   171f    373c
CRD$DONE_GENERAL_COMS  GlobRout   641 Lib01e    174f
CRD$FIND_PTABLE   ExtRout  Lib01   395c    543c    544c    545c    546c    547c    548c
CRD$GA_CRD_MESSAGE_BUFFER External Lib01  1100a   1163a   1171a
CRD$GA_CURRENT_DIAGNOSTIC External Lib01   255m    261a.   267.    274=    274a.   279.    286.
    Map         442
    Map         814   820a.   826.
    Map         865   871a.
    Map         925   931a.
    Map         977   983a.
    Map        1032  1035a.
    Map        1077  1080a.  1086.   1106a.  1108a.  1130a.  1180a.  1188a=  1191a=  1198.
    Map        1228
CRD$GA_FIRST_DIAGNOSTIC  External Lib01   254m    286.    382.    410.
CRD$GA_HDWR_SPRT_TABLE  External Lib01   541=    555=    568=    589=    601=    612=    623=    632=
CRD$GB_BASE_SYSTEM  External Lib01   540=    554=    567=    588=    600=    611=    622=    631=
CRD$GB_CRD_DEBUG  External Lib01   264.    276.    317.    663.   1083.   1195.
CRD$GET_GENERAL_COMMAND  ExtRout  Lib01   776c
CRD$GL_CURRENT_GENERAL_COM External Lib01   319.    321.    334=    334.    665.    667=
CRD$GL_CURRENT_STATE  External Lib01   287=    327=   1185=
CRD$GT_AUTOSIZER_NAME  External Lib01   726m    730.
CRD$GT_AUTOSIZER_SET_FLAGS External Lib01   448m    450.    455.
CRD$GT_DEV_UNAVAILABLE  External Lib01  1182a
CRD$GT_DS_LOAD_COM  External Lib01   723m    729.    829.
    Map         817   829.
CRD$GT_DS_SELECT_COM  External Lib01   868m    889.
CRD$GT_DS_SET_EVENT_COM  External Lib01   928m    941.
CRD$GT_DS_SET_FLAGS_COM  External Lib01   445m    454.    993.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Map         980   993.
CRD$GT_DS_START_COM  External Lib01  1029m   1042.   1234.
    Map        1231  1234.
CRD$GT_PASS   External Lib01   262a    693a
CRD$GT_RUNTIME   External Lib01  1100a   1100.
CRD$GT_TESTING_DEVICE  External Lib01  1100a   1100.
CRD$GT_TESTING_STARTED  External Lib01  1100a   1100.
CRD$INTERNAL_ERROR  ExtRout  Lib01   772c
CRD$K_ACTION_EQL_DO_NOTHING Literal    211
CRD$K_ACTION_RANGE_ERROR Literal    211
CRD$K_ADVANCE_DIAGNOSTIC Literal    211
CRD$K_ADVANCE_GENERAL_COM Literal    211
CRD$K_ADVANCE_STATE  Literal    211
CRD$K_ALLOCATION_ERROR  Literal    211
CRD$K_ASSIGN_PTABLE_PTRS Literal    211
CRD$K_AUTOSIZER_ERROR  Literal    211
CRD$K_AUTOSIZER_SET_FLAGS Literal    211
CRD$K_BAD_ACTION_NUMBER  Literal    211
CRD$K_BAD_TYPECODE_ERROR Literal    211
CRD$K_BASE_SYSTEM_IDC  Literal    211   554
CRD$K_BASE_SYSTEM_LEN  Literal    211   567
CRD$K_BASE_SYSTEM_NULL  Literal    211   540
CRD$K_BASE_SYSTEM_UDA_0  Literal    211   611     631
CRD$K_BASE_SYSTEM_UDA_1  Literal    211   600
CRD$K_BASE_SYSTEM_UDA_2  Literal    211   622
CRD$K_BASE_SYSTEM_UDA_3  Literal    211   588
CRD$K_CRD_INITIALIZATION Literal    211
CRD$K_CRD_MESSAGE_BUFFER_LEN Literal  Lib01  1100    1124
CRD$K_CREATE_HST  Literal    211
CRD$K_DATA_LENGTH_ERROR  Literal    211   772
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal    211   398     820     871     931     983    1035    1080
CRD$K_DDB_BLINK   Literal    211
CRD$K_DDB_FLINK   Literal    211   274     407
CRD$K_DDB_LAST_ENTRY  Literal    211
CRD$K_DDB_MEMORY_SIZE  Literal    211
CRD$K_DDB_MENU_DEVICE_NUMB Literal    211
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal    211
CRD$K_DDB_MENU_SUPPORT_SIZE Literal    211
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal    211
CRD$K_DDB_PTABLE_ENTRY  Literal    211   395    1106    1130    1180
CRD$K_DDB_STATUS  Literal    211   261    1108    1188    1191
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal    211
CRD$K_DEVICE_NAME  Literal    211   401     875
CRD$K_DIAGNOSTIC_ERROR  Literal    211
CRD$K_DIAGNOSTIC_NAME  Literal    211   824
CRD$K_DONE_GENERAL_COMS  Literal    211
CRD$K_DO_NOTHING  Literal    211
CRD$K_DS_COMMAND_SIZE  Literal  Lib01   462     737     770     779     784     837     897     949    1001    1050
          1241
CRD$K_DUMMY_10   Literal    211
CRD$K_DUMMY_11   Literal    211
CRD$K_DUMMY_12   Literal    211
CRD$K_DUMMY_13   Literal    211
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_DUMMY_3    Literal    211
CRD$K_DUMMY_4    Literal    211
CRD$K_DUMMY_5    Literal    211
CRD$K_DUMMY_6    Literal    211
CRD$K_DUMMY_7    Literal    211
CRD$K_DUMMY_8    Literal    211
CRD$K_DUMMY_9    Literal    211
CRD$K_EXIT_CRD    Literal    211
CRD$K_FAILURE    Literal  Lib01   637     773
CRD$K_HOOK_POINT  Literal    211
CRD$K_HS_INTERNAL_ERROR  Literal    211
CRD$K_IDC_EVDLB    Literal    211
CRD$K_IDC_EVDLC    Literal    211
CRD$K_IDC_EVDLD    Literal    211
CRD$K_IDC_EVRAD_0  Literal    211
CRD$K_IDC_EVRAD_1  Literal    211
CRD$K_IDC_LAST_DIAGNOSTIC Literal    211
CRD$K_INTERNAL_ERROR  Literal    211
CRD$K_LAST_ACTION_ROUTINE Literal    211
CRD$K_LAST_ENTRY  Literal    211
CRD$K_LAST_FUNCTION  Literal    211
CRD$K_LAST_HOOK_POINT  Literal    211
CRD$K_LAST_INTERNAL_ERROR Literal    211
CRD$K_LAST_TYPE    Literal    211
CRD$K_LESI_ENHCA  Literal    211
CRD$K_LESI_EVRMB_0  Literal    211
CRD$K_LESI_EVRMB_1  Literal    211
CRD$K_LESI_LAST_DIAGNOSTIC Literal    211
CRD$K_LEVEL_4_PASSED  Literal    211
CRD$K_LOAD_AUTOSIZER  Literal    211
CRD$K_LOAD_ERROR  Literal    211
CRD$K_LOAD_GENERAL_COM  Literal    211
CRD$K_LOAD_LOAD_COM  Literal    211
CRD$K_LOAD_SELECT_COM  Literal    211
CRD$K_LOAD_SET_EVENT_COM Literal    211
CRD$K_LOAD_SET_FLAGS_COM Literal    211
CRD$K_LOAD_START_COM  Literal    211
CRD$K_LOAD_SUP_FILE  Literal    211
CRD$K_MM_INTERNAL_ERROR  Literal    211
CRD$K_NEW_LINE    Literal    211
CRD$K_NUMB_GENERAL_COMMANDS Literal  Lib01   321
CRD$K_PREPARATION_ERROR  Literal    211
CRD$K_PREPARATION_ERROR_TEXT Literal    211
CRD$K_REPEAT_TURING  Literal  Lib01   289     337     412     451     668     695     882     938     990    1200
CRD$K_RUNTIME    Literal    211  1163    1171
CRD$K_SAME_LINE    Literal    211
CRD$K_SET_EVENT_ARGS  Literal    211   935
CRD$K_SET_FLAGS_ARGS  Literal    211   987
CRD$K_SET_UP_DIAGNOSTIC  Literal    211
CRD$K_START    Literal    211
CRD$K_START_AUTOSIZER  Literal    211
CRD$K_START_ERROR  Literal    211
CRD$K_START_QUALIFIERS  Literal    211  1039
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------------- -------- ----- ---------------
CRD$K_STAR_LINE   Literal    211
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal    211  1185
CRD$K_STATE_ADVANCE_GENERAL_COM Literal    211
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal    211
CRD$K_STATE_AUTOSIZER_ERROR Literal    211
CRD$K_STATE_AUTOSIZER_RUNNING Literal    211
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal    211
CRD$K_STATE_DIAGNOSTIC_ERROR Literal    211
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal    211
CRD$K_STATE_DONE_DIAGNOSTICS Literal    211   287
CRD$K_STATE_DONE_GENERAL_COMS Literal    211   327
CRD$K_STATE_DUMMY_1  Literal    211
CRD$K_STATE_DUMMY_10  Literal    211
CRD$K_STATE_DUMMY_12  Literal    211
CRD$K_STATE_DUMMY_13  Literal    211
CRD$K_STATE_DUMMY_2  Literal    211
CRD$K_STATE_DUMMY_3  Literal    211
CRD$K_STATE_DUMMY_4  Literal    211
CRD$K_STATE_DUMMY_6  Literal    211
CRD$K_STATE_DUMMY_7  Literal    211
CRD$K_STATE_DUMMY_8  Literal    211
CRD$K_STATE_EXIT_CRD  Literal    211
CRD$K_STATE_INTERNAL_ERROR Literal    211
CRD$K_STATE_LAST_STATE  Literal    211
CRD$K_STATE_LEVEL_4_PASSED Literal    211
CRD$K_STATE_LOAD_AUTOSIZER Literal    211
CRD$K_STATE_LOAD_ERROR  Literal    211
CRD$K_STATE_LOAD_GENERAL_COM Literal    211
CRD$K_STATE_LOAD_LOAD_COM Literal    211
CRD$K_STATE_LOAD_SELECT_COM Literal    211
CRD$K_STATE_LOAD_SET_EVENT_COM Literal    211
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal    211
CRD$K_STATE_LOAD_START_COM Literal    211
CRD$K_STATE_PREPARATION_ERROR Literal    211
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal    211
CRD$K_STATE_START  Literal    211
CRD$K_STATE_START_AUTOSIZER Literal    211
CRD$K_STATE_START_ERROR  Literal    211
CRD$K_SUCCESS   Unbound          462     737     784     837     897     949    1001    1050    1241
    Literal  Lib01   373     462     556     569     590     602     613     624     633     737
             784     837     897     949    1001    1050    1241
CRD$K_TEST_START_TEXT  Literal    211  1140    1171
CRD$K_TQ_INTERNAL_ERROR  Literal    211
CRD$K_TYPECODE   Literal    211
CRD$K_UDA_0_EVDLB  Literal    211
CRD$K_UDA_0_EVDLC  Literal    211
CRD$K_UDA_0_EVDLD  Literal    211
CRD$K_UDA_0_EVRLA_0  Literal    211
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal    211
CRD$K_UDA_1_EVDLB  Literal    211
CRD$K_UDA_1_EVDLC  Literal    211
CRD$K_UDA_1_EVDLD  Literal    211
CRD$K_UDA_1_EVRLA_0  Literal    211
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_UDA_1_EVRLA_1  Literal    211
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal    211
CRD$K_UDA_2_EVDLB  Literal    211
CRD$K_UDA_2_EVDLC  Literal    211
CRD$K_UDA_2_EVDLD  Literal    211
CRD$K_UDA_2_EVRLA_0  Literal    211
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal    211
CRD$K_UDA_3_EVDLB  Literal    211
CRD$K_UDA_3_EVDLC  Literal    211
CRD$K_UDA_3_EVDLD  Literal    211
CRD$K_UDA_3_EVRLA_0  Literal    211
CRD$K_UDA_3_EVRLA_1  Literal    211
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal    211
CRD$LEVEL_4_PASSED  GlobRout   673 Lib01e    177f
CRD$LOAD_AUTOSIZER  GlobRout   699 Lib01e    181f
CRD$LOAD_GENERAL_COM  GlobRout   741 Lib01e    184f
CRD$LOAD_LOAD_COM  GlobRout   790 Lib01e    186f
CRD$LOAD_SELECT_COM  GlobRout   841 Lib01e    189f
CRD$LOAD_SET_EVENT_COM  GlobRout   901 Lib01e    192f
CRD$LOAD_SET_FLAGS_COM  GlobRout   953 Lib01e    195f
CRD$LOAD_START_COM  GlobRout  1005 Lib01e    198f
CRD$PRINT   External Lib01   262ac   265ac   277ac   319ac   665ac   693ac  1084ac  1163ac  1171ac  1182ac
          1196ac
CRD$PRINT_DDB   ExtRout  Lib01   267c    279c    405c    826c   1086c   1198c
CRD$PRINT_DS_COMMAND  ExtRout  Lib01   460c    735c    782c    835c    895c    947c    999c   1048c   1239c
CRD$SET_UP_DIAGNOSTIC  GlobRout  1054 Lib01e    201f
CRD$START_AUTOSIZER  GlobRout  1204 Lib01e    206f
CRD$STOP   ExtRout  Lib01   375c
CSR    Local      537   563=    564.    577=    578.
DATA_BUFFER_ADDRESS  RoutForm   741   772.    779a=   782.
DATA_MAX_LENGTH   RoutForm   741   770.    772.    782.
DDB$V_STATUS_CHECK_DEVICE_TYPE Macro    Lib01  1108
DDB$V_STATUS_DEV_TEST_COMPLETE Macro    Lib01  1191
DDB$V_STATUS_MESSAGE_PRINTED Macro    Lib01   261    1188
DDB_PTR    Local      368   382=    389m    395a=   398a.   405.    407=    407a.   410.
DEVICE_DATA_BLOCK_STRUCTURE Structur Lib01   254     255     389     442     814     865     925     977    1032    1077
          1228
DEVICE_NAME   Bind       875   877.    890.
DIAG_NAME   Bind       824   830.
DIAG_SUPPORT_VECTOR  Bind       398   401.
DIAG_VECTOR_PTR   Bind       820   824.
    Bind       871   875.
    Bind       931   935.
    Bind       983   987.
    Bind      1035  1039.
    Bind      1080  1140.   1163.   1171.
DS$K_PRINTB   Literal    210
    Literal    262
    Literal    265
    Literal    277
    Literal    319
    Literal    665
    Literal    693
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal   1084
    Literal   1163
    Literal   1171
    Literal   1182
    Literal   1196
DS$K_PRINTF   Literal    210
    Literal    262   262
    Literal    265   265
    Literal    277   277
    Literal    319   319
    Literal    665   665
    Literal    693   693
    Literal   1084  1084
    Literal   1163  1163
    Literal   1171  1171
    Literal   1182  1182
    Literal   1196  1196
DS$K_PRINTI   Literal    210
    Literal    262
    Literal    265
    Literal    277
    Literal    319
    Literal    665
    Literal    693
    Literal   1084
    Literal   1163
    Literal   1171
    Literal   1182
    Literal   1196
DS$K_PRINTX   Literal    210
    Literal    262
    Literal    265
    Literal    277
    Literal    319
    Literal    665
    Literal    693
    Literal   1084
    Literal   1163
    Literal   1171
    Literal   1182
    Literal   1196
DS$K_TYPE_ABORT_PROGRAM  Literal    210
    Literal    262
    Literal    265
    Literal    277
    Literal    319
    Literal    665
    Literal    693
    Literal   1084
    Literal   1163
    Literal   1171
    Literal   1182
    Literal   1196
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DS$K_TYPE_ABORT_TEST  Literal    210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_COMMAND_ERR  Literal    210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_COMMAND_OUT  Literal    210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_CRD_AUTOTEST  Literal    210
    Literal     262    262
    Literal     265    265
    Literal     277    277
    Literal     319    319
    Literal     665    665
    Literal     693    693
    Literal    1084   1084
    Literal    1163   1163
    Literal    1171   1171
    Literal    1182   1182
    Literal    1196   1196
DS$K_TYPE_DS_PROMPT  Literal    210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_DS_START  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_ERRDEV  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_ERRHARD  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_ERROR_BODY  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
```

```
Symbol                            Type  Defined  Referenced ...
-------------------------------- -------- ----- ---------------
    Literal    1182
    Literal    1196
DS$K_TYPE_ERROR_END  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_ERRPREP  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_ERRSOFT  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_ERRSUP  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_ERRSYS  Literal     210
    Literal     262
    Literal     265
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_ERR_HALT  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_EXCEPTION  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_EXCEPTION_HEAD Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_FIRST_PASS  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_GENERAL  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_GENERAL_ERROR  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_NO_TESTS  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_PARAM_ERROR  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_PROGRAM_END  Literal     210
```

```
Symbol                            Type  Defined  Referenced ...
-------------------------------- -------- ----- ---------------
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_PROGRAM_INFO  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_PROGRAM_START  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_QIO_INVADP  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_QIO_NODRIVER  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_QIO_WRONGVER  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_SCRIPT_ECHO  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_SCRIPT_PNF  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_SCRIPT_PROMPT  Literal     210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal   1196
DS$K_TYPE_SCRIPT_SKIP  Literal    210
    Literal    262
    Literal    265
    Literal    277
    Literal    319
    Literal    665
    Literal    693
    Literal   1084
    Literal   1163
    Literal   1171
    Literal   1182
    Literal   1196
DS$K_TYPE_SEQUENCE_ERROR Literal    210
    Literal    262
    Literal    265
    Literal    277
    Literal    319
    Literal    665
    Literal    693
    Literal   1084
    Literal   1163
    Literal   1171
    Literal   1182
    Literal   1196
DS$K_TYPE_START_ERR  Literal    210
    Literal    262
    Literal    265
    Literal    277
    Literal    319
    Literal    665
    Literal    693
    Literal   1084
    Literal   1163
    Literal   1171
    Literal   1182
    Literal   1196
DS$K_TYPE_START_LIST  Literal    210
    Literal    262
    Literal    265
    Literal    277
    Literal    319
    Literal    665
    Literal    693
    Literal   1084
    Literal   1163
    Literal   1171
    Literal   1182
    Literal   1196
DS$K_TYPE_SUMMARY  Literal    210
    Literal    262
    Literal    265
    Literal    277
```

```
Symbol                          , Type  Defined  Referenced ...
------------------------------- -------- ----- ---------------
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
DS$K_TYPE_USER_PROMPT  Literal    210
    Literal     262
    Literal     265
    Literal     277
    Literal     319
    Literal     665
    Literal     693
    Literal    1084
    Literal    1163
    Literal    1171
    Literal    1182
    Literal    1196
FALSE    Literal  Lib01   527     528     529     531     533     535    1191
GET1ST_     Macro    Lib04   210     211     262     265     277     319     665     693    1084    1163
            1171    1182    1196
GET2ND_     Unbound          210     211     262     265     277     319     665     693    1084    1163
            1171    1182    1196
HARDWARE_SUPPORT_VECTOR  Structur Lib01   399     821     872     932     984    1036    1081
HP$A_DEVICE   Field    Lib03   563.    577.
HP$K_LENGTH   Literal  Lib03   538    1133
HP$T_TYPE   Field    Lib03  1136a
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal    211
HS$K_ACTION_ADVANCE_GENERAL_COM Literal    211
HS$K_ACTION_ADVANCE_STATE Literal    211
HS$K_ACTION_CHANGE_TABLE Literal    211
HS$K_ACTION_DIAGNOSTIC_ERROR Literal    211
HS$K_ACTION_DONE_GENERAL_COMS Literal    211
HS$K_ACTION_DO_NOTHING  Literal    211
HS$K_ACTION_DUMMY_3  Literal    211
HS$K_ACTION_DUMMY_4  Literal    211
HS$K_ACTION_DUMMY_5  Literal    211
HS$K_ACTION_DUMMY_6  Literal    211
HS$K_ACTION_INIT_HS  Literal    211
HS$K_ACTION_INTERNAL_ERROR Literal    211
HS$K_ACTION_LAST_ACTION  Literal    211
HS$K_ACTION_LOAD_ERROR  Literal    211
HS$K_ACTION_LOAD_GENERAL_COM Literal    211
HS$K_ACTION_LOAD_LOAD_COM Literal    211
HS$K_ACTION_LOAD_SELECT_COM Literal    211
HS$K_ACTION_LOAD_SET_EVENT_COM Literal    211
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal    211
HS$K_ACTION_LOAD_START_COM Literal    211
HS$K_ACTION_PREPARATION_ERROR Literal    211
HS$K_ACTION_START_ERROR  Literal    211
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal    211
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_STATE_ADVANCE_GENERAL_COM Literal    211
HS$K_STATE_CHANGE_TABLE  Literal    211
HS$K_STATE_DIAGNOSTIC_ERROR Literal    211
HS$K_STATE_DIAGNOSTIC_RUNNING Literal    211
HS$K_STATE_DONE_GENERAL_COMS Literal    211
HS$K_STATE_DUMMY_1  Literal    211
HS$K_STATE_DUMMY_2  Literal    211
HS$K_STATE_DUMMY_3  Literal    211
HS$K_STATE_DUMMY_4  Literal    211
HS$K_STATE_DUMMY_6  Literal    211
HS$K_STATE_INIT_HS  Literal    211
HS$K_STATE_INTERNAL_ERROR Literal    211
HS$K_STATE_LAST_STATE  Literal    211
HS$K_STATE_LOAD_ERROR  Literal    211
HS$K_STATE_LOAD_GENERAL_COM Literal    211
HS$K_STATE_LOAD_LOAD_COM Literal    211
HS$K_STATE_LOAD_SELECT_COM Literal    211
HS$K_STATE_LOAD_SET_EVENT_COM Literal    211
HS$K_STATE_LOAD_SET_FLAGS_COM Literal    211
HS$K_STATE_LOAD_START_COM Literal    211
HS$K_STATE_PREPARATION_ERROR Literal    211
HS$K_STATE_START_ERROR  Literal    211
IDC_DEVICE_NAME   Bind       517   543a
IDC_PTABLE   Register   527   527=    543=    551.
LENGTH    Bind      1100  1100
LESI_DEVICE_NAME  Bind       518   544a
LESI_PTABLE   Register   528   528=    544=    559.    562.
MEN$K_HS_STATE_TABLE  Literal    211
MEN$K_MM_STATE_TABLE  Literal    211
MEN$K_TQ_STATE_TABLE  Literal    211
MENU$K_EXIT_AUTOSIZER_ERROR Literal    211
MENU$K_EXIT_INTERNAL_ERROR Literal    211
MENU$K_EXIT_LAST_REASON  Literal    211
MENU$K_EXIT_OPERATOR_ABORT Literal    211
MENU$K_EXIT_OPERATOR_STOP Literal    211
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal    211
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal    211
MM$K_ACTION_ADVANCE_STATE Literal    211
MM$K_ACTION_AUTOSIZER_ERROR Literal    211
MM$K_ACTION_BUILD_DDBS  Literal    211
MM$K_ACTION_BUILD_HSTS  Literal    211
MM$K_ACTION_CHANGE_TABLE Literal    211
MM$K_ACTION_DONE_INITS  Literal    211
MM$K_ACTION_DO_NOTHING  Literal    211
MM$K_ACTION_DUMMY_3  Literal    211
MM$K_ACTION_DUMMY_4  Literal    211
MM$K_ACTION_DUMMY_5  Literal    211
MM$K_ACTION_DUMMY_6  Literal    211
MM$K_ACTION_DUMMY_7  Literal    211
MM$K_ACTION_DUMMY_8  Literal    211
MM$K_ACTION_INTERNAL_ERROR Literal    211
MM$K_ACTION_LAST_ACTION  Literal    211
MM$K_ACTION_LOAD_AUTOSIZER Literal    211
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- --------------
MM$K_ACTION_MENU_PROMPT  Literal    211
MM$K_ACTION_PRINT_MENU  Literal    211
MM$K_ACTION_PRINT_MENU_HEADING Literal    211
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal    211
MM$K_ACTION_START_AUTOSIZER Literal    211
MM$K_STATE_AUTOSIZER_ERROR Literal    211
MM$K_STATE_AUTOSIZER_RUNNING Literal    211
MM$K_STATE_BUILD_DDBS  Literal    211
MM$K_STATE_BUILD_HSTS  Literal    211
MM$K_STATE_CHANGE_TABLE  Literal    211
MM$K_STATE_DONE_INITS  Literal    211
MM$K_STATE_DUMMY_1  Literal    211
MM$K_STATE_DUMMY_2  Literal    211
MM$K_STATE_DUMMY_3  Literal    211
MM$K_STATE_DUMMY_5  Literal    211
MM$K_STATE_DUMMY_7  Literal    211
MM$K_STATE_DUMMY_8  Literal    211
MM$K_STATE_INTERNAL_ERROR Literal    211
MM$K_STATE_LAST_STATE  Literal    211
MM$K_STATE_LOAD_AUTOSIZER Literal    211
MM$K_STATE_MENU_PROMPT  Literal    211
MM$K_STATE_PRINT_MENU  Literal    211
MM$K_STATE_PRINT_MENU_HEADING Literal    211
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal    211
MM$K_STATE_START  Literal    211
MM$K_STATE_START_AUTOSIZER Literal    211
MODNAM    Macro    Lib02   227
MOVC3    Builtin   1144  1146    1155
NUL2ND_    Macro    Lib04   210     211     262     265     277     319     665     693    1084    1163
             1171    1182    1196
PTABLE    Bind      1133  1136a
PTABLE_DEVICE_TYPE  Bind      1136  1152.   1155a
PTABLE_DEVICE_TYPE_LENGTH Local     1120  1152=   1155.
PTABLE_PTR   Local      538   562=    563a.   576=    577a.
    Bind      1130  1133a
RA60_DRIVE_0_DEVICE_NAME Bind       521   546a
RA60_DRIVE_0_PTABLE  Register   531   531=    546=    585.    619.
RA60_DRIVE_1_DEVICE_NAME Bind       525   548a
RA60_DRIVE_1_PTABLE  Register   535   535=    548=    585.    596.    608.    619.
RA8081_DRIVE_0_DEVICE_NAME Bind       523   547a
RA8081_DRIVE_0_PTABLE  Register   533   533=    547=    597.    608.
SET_EVENT_ARGS   Bind       935   937.    942.
SET_FLAGS_ARGS   Bind       987   989.    994.
START_QUALIFIERS  Bind      1039  1043.
STRING    Bind      1100  1100.
    Bind      1100  1100.
    Bind      1100  1100.
    Bind      1100  1100.
    Bind      1100  1100.
TEMP_TEST_START_TEXT  Local     1124  1146=   1155=   1163a
TEST_START_TEXT   Bind      1140  1146.   1146a
TQ$K_ACTION_ADVANCE_STATE Literal    211
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
TQ$K_ACTION_CHANGE_TABLE Literal    211
TQ$K_ACTION_DONE_TEST_QUEUE Literal    211
TQ$K_ACTION_DO_NOTHING  Literal    211
TQ$K_ACTION_DUMMY_3  Literal    211
TQ$K_ACTION_DUMMY_4  Literal    211
TQ$K_ACTION_DUMMY_5  Literal    211
TQ$K_ACTION_GET_DDB  Literal    211
TQ$K_ACTION_INIT_TQ  Literal    211
TQ$K_ACTION_INTERNAL_ERROR Literal    211
TQ$K_ACTION_LAST_ACTION  Literal    211
TQ$K_STATE_CHANGE_TABLE  Literal    211
TQ$K_STATE_DONE_TEST_QUEUE Literal    211
TQ$K_STATE_DUMMY_1  Literal    211
TQ$K_STATE_DUMMY_2  Literal    211
TQ$K_STATE_DUMMY_3  Literal    211
TQ$K_STATE_DUMMY_4  Literal    211
TQ$K_STATE_DUMMY_5  Literal    211
TQ$K_STATE_GET_DDB  Literal    211
TQ$K_STATE_INIT_TQ  Literal    211
TQ$K_STATE_INTERNAL_ERROR Literal    211
TQ$K_STATE_LAST_STATE  Literal    211
TRUE    Literal  Lib01  1188
UDA_DEVICE_NAME   Bind       519   545a
UDA_PTABLE   Register   529   529=    545=    573.    576.


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDACT2.B32;1
    2  Module CRDACT2
  146  Library #1 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
  149  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
  152  Library #3 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
  155  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
 1244  Eludom CRDACT2


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     a Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration
```

```
;    Library Statistics
;
;          -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486         69         14         62          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0         46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923         18          1         94          00:00.2
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3          0       1000          00:04.3




;     COMMAND QUALIFIERS

; BLISS CRDACT2.B32/LIST=CRDACT2.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDACT2.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  1689 code + 292 data bytes
; Run Time:    00:54.4
; Elapsed Time:    01:25.2
; Lines/CPU Min:    1372
; Lexemes/CPU-Min: 58965
; Memory Used: 306 pages
; Compilation Complete
```

```
;    0001  0 %TITLE '*** CRDCTRLC Module'
;    0002  0 MODULE CRDCTRLC (    ! Handle the DS$GL_Flags bits pertaining to ^C
;    0003  0   IDENT = '01-00'
;    0004  0   ) =
;    0005  0 !++
;    0006  0 ! COPYRIGHT (c) 1980, 1981
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! Handle setting and clearing the bits in the DS$GL_Flags longword.
;    0032  0 !
;    0033  0 ! Author:
;    0034  0 !
;    0035  0 ! Jack Stansbury
;    0036  0 !
;    0037  0 ! Creation Date:
;    0038  0 !
;    0039  0 ! July 29, 1982
;    0040  0 !
;    0041  0 ! Version:
;    0042  0 !
;    0043  0 ! 6.9
;    0044  0 !
;    0045  0 ! Modified By:
;    0046  0 !
;    0047  0 ! None
;    0048  0 !
```

```
;    0049  0 %SBTTL 'Libraries'
;    0050  1 BEGIN     ! Begin the CRDCTRLC module
;    0051  1
;    0052  1 LIBRARY
;    0053  1  '$DS';    ! Use Ds.L32 first
;    0054  1
;    0055  1 LIBRARY
;    0056  1  '$Diag';   ! Use Diag.L32 second
;    0057  1
;    0058  1 LIBRARY
;    0059  1  '$CRD';    ! Use CRD.L32 third
;    0060  1
;    0061  1 LIBRARY
;    0062  1  'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0063  1 %SBTTL 'Table of Contents'
;    0064  1
;    0065  1 FORWARD ROUTINE
;    0066  1  CRD$Set_Ctrl_C_Handlers  : JSB_R0_R1   NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0067  1      ! Establish the two control-c handlers for the DS (for CRD)
;    0068  1
;    0069  1  CRD$Clear_Ctrl_C_Handlers : JSB_NO_REGS NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0070  1      ! Clear the two handler addresses
;    0071  1
;    0072  1  CRD$Enable_Ctrl_C  : JSB_NO_REGS NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0073  1      ! Enable catching of ^C's.
;    0074  1
;    0075  1  CRD$Disable_Ctrl_C  : JSB_NO_REGS NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0076  1      ! Disable catching of ^C's - i.e., ignore them
;    0077  1
;    0078  1  CRD$Check_Ctrl_C  : JSB_NO_REGS         ADDRESSING_MODE (LONG_RELATIVE);
;    0079  1      ! Return the setting of the $DS_GL_Flags <DS$V_CtrlC> bit.
```

```
;    0080  1 %SBTTL 'Included Macros and Literals'
;    0081  1
;    0082  1  $CRD_Literals;    ! The CRD literals
```

```
;    0083  1 %SBTTL 'Psect Definitions'
;    0084  1
;    0085  1 PSECT
;    0086  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0087  1
;    0088  1 PSECT
;    0089  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0090  1
;    0091  1 PSECT
;    0092  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0093  1
;    0094  1 PSECT
;    0095  1     GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0096  1 %SBTTL 'Module-Global Static Storage'
;    0097  1
;    0098  1  ModNam ('CRDCTRLC');  ! Module name for ErrSup macro
```

```
;    0099  1 %SBTTL 'CRD$Set_Ctrl_C_Handlers Routine'
;    0100  1
;    0101  1 GLOBAL ROUTINE CRD$Set_Ctrl_C_Handlers (First_Handler, Second_Handler) : JSB_R0_R1 NOVALUE
;    0102  1
;    0103  1 !++
;    0104  1 ! Functional Description:
;    0105  1 !
;    0106  1 ! Set up two ^C handler routines.
;    0107  1 !
;    0108  1 ! Formal Input Parameters:
;    0109  1 !
;    0110  1 ! First_Handler: The first ^C handler routine address.
;    0111  1 ! Second_Handler: The second ^C handler routine address.
;    0112  1 !
;    0113  1 ! Formal Output Parameters:
;    0114  1 !
;    0115  1 ! None
;    0116  1 !
;    0117  1 ! Side Effects:
;    0118  1 !
;    0119  1 ! None
;    0120  1 !
;    0121  1 ! Completion Codes:
;    0122  1 !
;    0123  1 ! None
;    0124  1 !--
```

```
;    0125  2 BEGIN     ! Routine CRD$Set_Ctrl_C_Handlers
;    0126  2  .CRD$GA_DS_Ctrl_C_First = .First_Handler;
;    0127  2       ! Set up first handler
;    0128  2  .CRD$GA_DS_ctrl_C_Second = .Second_Handler;
;    0129  2       ! Set up second handler
;    0130  2  RETURN
;    0131  1 END;     ! Routine CRD$Set_Ctrl_C_Handlers


            .TITLE  CRDCTRLC *** CRDCTRLC Module
            .IDENT  \01-00\

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

   43  4C  52  54  43  44  52  43  08  00000 P.AAA:  .ASCII  <8>\CRDCTRLC\          ;

           $MODULE=       P.AAA
            .EXTRN  CRD$SET_CTRL_C_HANDLERS
            .EXTRN  CRD$CLEAR_CTRL_C_HANDLERS
            .EXTRN  CRD$ENABLE_CTRL_C
            .EXTRN  CRD$DISABLE_CTRL_C
            .EXTRN  CRD$CHECK_CTRL_C
            .EXTRN  CRD$GA_DS_CTRL_C_FIRST
            .EXTRN  CRD$GA_DS_CTRL_C_SECOND

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

      00000000G  FF      50  D0 00000 CRD$SET_CTRL_C_HANDLERS::
          MOVL    FIRST_HANDLER, @CRD$GA_DS_CTRL_C_FIRST      ; 0126
      00000000G  FF      51  D0 00007    MOVL    SECOND_HANDLER, @CRD$GA_DS_CTRL_C_SECOND    ; 0128
       05 0000E    RSB                 ; 0131

; Routine Size:  15 bytes,    Routine Base:  CODE + 0000
```

```
;    0132  1 %SBTTL 'CRD$Clear_Ctrl_C_Handlers Routine'
;    0133  1
;    0134  1 GLOBAL ROUTINE CRD$Clear_Ctrl_C_Handlers : JSB_NO_REGS NOVALUE =
;    0135  1
;    0136  1 !++
;    0137  1 ! Functional Description:
;    0138  1 !
;    0139  1 ! Clear both of the DS ^C handlers.
;    0140  1 !
;    0141  1 ! Formal Input Parameters:
;    0142  1 !
;    0143  1 ! None
;    0144  1 !
;    0145  1 ! Formal Output Parameters:
;    0146  1 !
;    0147  1 ! None
;    0148  1 !
;    0149  1 ! Side Effects:
;    0150  1 !
;    0151  1 ! None
;    0152  1 !
;    0153  1 ! Completion Codes:
;    0154  1 !
;    0155  1 ! None
;    0156  1 !--
```

```
;    0157  2 BEGIN     ! Routine CRD$Clear_Ctrl_C_Handlers
;    0158  2  .CRD$GA_DS_Ctrl_C_First  = 0;
;    0159  2  .CRD$GA_DS_Ctrl_C_Second = 0;
;    0160  2  RETURN
;    0161  1 END;     ! Routine CRD$Clear_Ctrl_C_Handlers



        00000000G  FF  D4 00000 CRD$CLEAR_CTRL_C_HANDLERS::
            CLRL    @CRD$GA_DS_CTRL_C_FIRST        ; 0158
        00000000G  FF  D4 00006    CLRL    @CRD$GA_DS_CTRL_C_SECOND        ; 0159
         05 0000C    RSB                 ; 0161

; Routine Size:  13 bytes,    Routine Base:  CODE + 000F
```

```
;    0162  1 %SBTTL 'CRD$Enable_Ctrl_C Routine'
;    0163  1
;    0164  1 GLOBAL ROUTINE CRD$Enable_Ctrl_C : JSB_NO_REGS NOVALUE =
;    0165  1
;    0166  1 !++
;    0167  1 ! Functional Description:
;    0168  1 !
;    0169  1 ! Enable the DisablCC bit in the DS$GL_Flags longword. Th's enables ^C's to be recognized (i.e., not
;    0170  1 ! ignored) by the DS (and CRD).
;    0171  1 !
;    0172  1 ! Formal Input Parameters:
;    0173  1 !
;    0174  1 ! None
;    0175  1 !
;    0176  1 ! Formal Output Parameters:
;    0177  1 !
;    0178  1 ! None
;    0179  1 !
;    0180  1 ! Side Effects:
;    0181  1 !
;    0182  1 ! None
;    0183  1 !
;    0184  1 ! Completion Codes:
;    0185  1 !
;    0186  1 ! None
;    0187  1 !
```

```
;    0188  2 BEGIN     ! Routine CRD$Enable_Ctrl_C
;    0189  2  BIND
;    0190  2   DS$GL_Flags = (.CRD$GA_DS_Flags);
;    0191  2
;    0192  2  DS$GL_Flags <DS$V_DisablCC> = False;
;    0193  2       ! Enable ^C's
;    0194  2  RETURN
;    0195  1 END;     ! Routine CRD$Enable_Ctrl_C



            .EXTRN  CRD$GA_DS_FLAGS

00000000G  FF           01        18      00  F0 00000 CRD$ENABLE_CTRL_C::
           INSV    #0, #24, #1, @CRD$GA_DS_FLAGS        ; 0192
        05 00009    RSB                 ; 0195

; Routine Size:  10 bytes,    Routine Base:  CODE + 001C
```

```
;    0196  1 %SBTTL 'CRD$Disable_Ctrl_C Routine'
;    0197  1
;    0198  1 GLOBAL ROUTINE CRD$Disable_Ctrl_C : JSB_NO_REGS NOVALUE =
;    0199  1
;    0200  1 !++
;    0201  1 ! Functional Description:
;    0202  1 !
;    0203  1 ! Disable catching of ^C's. That is, ignore them if they are typed.
;    0204  1 !
;    0205  1 ! Formal Input Parameters:
;    0206  1 !
;    0207  1 ! None
;    0208  1 !
;    0209  1 ! Formal Output Parameters:
;    0210  1 !
;    0211  1 ! None
;    0212  1 !
;    0213  1 ! Side Effects:
;    0214  1 !
;    0215  1 ! None
;    0216  1 !
;    0217  1 ! Completion Codes:
;    0218  1 !
;    0219  1 ! None
;    0220  1 !--
```

```
;    0221  2 BEGIN     ! Routine CRD$Disable_Ctrl_C
;    0222  2  BIND
;    0223  2   DS$GL_Flags = (.CRD$GA_DS_Flags);
;    0224  2
;    0225  2  DS$GL_Flags <DS$V_DisablCC> = True;
;    0226  2       ! Disable ^C's
;    0227  2  RETURN
;    0228  1 END;     ! Routine CRD$Disable_Ctrl_C




00000000G  FF          01        18      01  F0 00000 CRD$DISABLE_CTRL_C::
          INSV    #1, #24, #1, @CRD$GA_DS_FLAGS        ; 0225
       05 00009    RSB                 ; 0228

; Routine Size:  10 bytes,    Routine Base:  CODE + 0026
```

```
;    0229  1 %SBTTL 'CRD$Set_Ctrl_C_Flag Routine'
;    0230  1
;    0231  1 GLOBAL ROUTINE CRD$Set_Ctrl_C_Flag : JSB_NO_REGS NOVALUE =
;    0232  1
;    0233  1 !++
;    0234  1 ! Functional Description:
;    0235  1 !
;    0236  1 ! Set the DS$GL_Flags <DS$V_CtrlC> bit.
;    0237  1 !
;    0238  1 ! Formal Input Parameters:
;    0239  1 !
;    0240  1 ! None
;    0241  1 !
;    0242  1 ! Formal Output Parameters:
;    0243  1 !
;    0244  1 ! None
;    0245  1 !
;    0246  1 ! Side Effects:
;    0247  1 !
;    0248  1 ! None
;    0249  1 !
;    0250  1 ! Completion Codes:
;    0251  1 !
;    0252  1 ! None
;    0253  1 !--
```

```
;    0254  2 BEGIN      ! Routine CRD$Set_Ctrl_C_Flag
;    0255  2  BIND
;    0256  2   DS$GL_Flags = (.CRD$GA_DS_Flags);
;    0257  2
;    0258  2  DS$GL_Flags <DS$V_CtrlC> = True;
;    0259  2       ! Set the bit, indicating a ^C was typed
;    0260  2  RETURN
;    0261  1 END;      ! Routine CRD$Set_Ctrl_C_Flag



            .EXTRN  CRD$SET_CTRL_C_FLAG

        00000000G  FF       01  88 00000 CRD$SET_CTRL_C_FLAG::
            BISB2    #1, @CRD$GA_DS_FLAGS          ; 0258
         05 00007    RSB                 ; 0261

; Routine Size:  8 bytes,    Routine Base:  CODE + 0030
```

```
;    0262  1 %SBTTL 'CRD$Clear_Ctrl_C_Flag Routine'
;    0263  1
;    0264  1 GLOBAL ROUTINE CRD$Clear_Ctrl_C_Flag : JSB_NO_REGS NOVALUE =
;    0265  1
;    0266  1 !++
;    0267  1 ! Functional Description:
;    0268  1 !
;    0269  1 ! Clear the DS$GL_Flags <DS$V_CtrlC> bit.
;    0270  1 !
;    0271  1 ! Formal Input Parameters:
;    0272  1 !
;    0273  1 ! None
;    0274  1 !
;    0275  1 ! Formal Output Parameters:
;    0276  1 !
;    0277  1 ! None
;    0278  1 !
;    0279  1 ! Side Effects:
;    0280  1 !
;    0281  1 ! None
;    0282  1 !
;    0283  1 ! Completion Codes:
;    0284  1 !
;    0285  1 ! None
;    0286  1 !--
```

```
;    0287  2 BEGIN     ! Routine CRD$Clear_Ctrl_C_Flag
;    0288  2  BIND
;    0289  2   DS$GL_Flags = (.CRD$GA_DS_Flags);
;    0290  2
;    0291  2  DS$GL_Flags <DS$V_CtrlC> = False;
;    0292  2       ! Clear the bit, so that it appears no ^C was typed.
;    0293  2  RETURN
;    0294  1 END;     ! Routine CRD$Clear_Ctrl_C_Flag



            .EXTRN  CRD$CLEAR_CTRL_C_FLAG

        00000000G  FF       01  8A 00000 CRD$CLEAR_CTRL_C_FLAG::
            BICB2   #1, @CRD$GA_DS_FLAGS          ; 0291
         05 00007    RSB                 ; 0294

; Routine Size:  8 bytes,    Routine Base:  CODE + 0038
```

```
;    0295  1 %SBTTL 'CRD$Check_Ctrl_C Routine'
;    0296  1
;    0297  1 GLOBAL ROUTINE CRD$Check_Ctrl_C : JSB_NO_REGS =
;    0298  1
;    0299  1 !++
;    0300  1 ! Functional Description:
;    0301  1 !
;    0302  1 ! Return the setting of the $DS_GL_Flags <DS$V_CtrlC> bit.
;    0303  1 !
;    0304  1 ! Formal Input Parameters:
;    0305  1 !
;    0306  1 ! None
;    0307  1 !
;    0308  1 ! Formal Output Parameters:
;    0309  1 !
;    0310  1 ! None
;    0311  1 !
;    0312  1 ! Side Effects:
;    0313  1 !
;    0314  1 ! None
;    0315  1 !
;    0316  1 ! Completion Codes:
;    0317  1 !
;    0318  1 ! True  - bit was set
;    0319  1 ! False - bit was clear
;    0320  1 !--
```

```
;   0321  2 BEGIN      ! Routine CRD$Check_Ctrl_C
;   0322  2  BIND
;   0323  2    DS$GL_Flags = (.CRD$GA_DS_Flags);
;   0324  2
;   0325  2  RETURN (.DS$GL_Flags <DS$V_CtrlC>);
;   0326  1 END;      ! Routine CRD$Check_Ctrl_C




    50 00000000G  FF         01      00  EF 00000 CRD$CHECK_CTRL_C::
          EXTZV    #0, #1, @CRD$GA_DS_FLAGS, R0        ; 0325
       05 00009    RSB                 ; 0326

; Routine Size:  10 bytes,    Routine Base:  CODE + 0040
```

```
;    0327  1 END      ! End of CRDCTRLC module
;    0328  0 ELUDOM



;            PSECT SUMMARY
;
; Name     Bytes          Attributes
;
;  DATA            9  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE           74  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



Symbol                              Type  Defined  Referenced ...
------------------------------ ------- - -- ---------------
$CRD_LITERALS   Macro    Lib03    82
$EQULST    Macro    Lib04    82
$ER    Comptime    98
$MODULE    Bind        98
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     82
AUTO$K_EXIT_CONTROL_C  Literal     82
AUTO$K_EXIT_DUMMY_2  Literal     82
AUTO$K_EXIT_DUMMY_3  Literal     82
AUTO$K_EXIT_ERRORS_COMPLETE Literal     82
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     82
AUTO$K_EXIT_INTERNAL_ERROR Literal     82
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     82
AUTO$K_EXIT_LAST_REASON  Literal     82    82
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     82
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     82
AUTO$K_EXIT_NOERRORS_PREP Literal     82
CRD$CHECK_CTRL_C  GlobRout   297 Lib03e     78f
CRD$CLEAR_CTRL_C_FLAG  GlobRout   264 Lib03e
CRD$CLEAR_CTRL_C_HANDLERS GlobRout   134 Lib03e     69f
CRD$DISABLE_CTRL_C  GlobRout   198 Lib03e     75f
CRD$ENABLE_CTRL_C  GlobRout   164 Lib03e     72f
CRD$GA_DS_CTRL_C_FIRST  External Lib03   126a=   158a=
CRD$GA_DS_CTRL_C_SECOND  External Lib03   128a=   159a=
CRD$GA_DS_FLAGS   External Lib03   190.    223.    256.    289.    323.
CRD$K_ACTION_EQL_DO_NOTHING Literal     82
CRD$K_ACTION_RANGE_ERROR Literal     82
CRD$K_ADVANCE_DIAGNOSTIC Literal     82
CRD$K_ADVANCE_GENERAL_COM Literal     82
CRD$K_ADVANCE_STATE  Literal     82
CRD$K_ALLOCATION_ERROR  Literal     82
CRD$K_ASSIGN_PTABLE_PTRS Literal     82
CRD$K_AUTOSIZER_ERROR  Literal     82
CRD$K_AUTOSIZER_SET_FLAGS Literal     82
CRD$K_BAD_ACTION_NUMBER  Literal     82
```

Symbol	Type	Defined	Referenced ...
CRD$K_BAD_TYPECODE_ERROR	Literal		82
CRD$K_BASE_SYSTEM_IDC	Literal		82
CRD$K_BASE_SYSTEM_LESI	Literal		82
CRD$K_BASE_SYSTEM_NULL	Literal		82
CRD$K_BASE_SYSTEM_UDA_0	Literal		82
CRD$K_BASE_SYSTEM_UDA_1	Literal		82
CRD$K_BASE_SYSTEM_UDA_2	Literal		82
CRD$K_BASE_SYSTEM_UDA_3	Literal		82
CRD$K_CRD_INITIALIZATION	Literal		82
CRD$K_CREATE_HST	Literal		82
CRD$K_DATA_LENGTH_ERROR	Literal		82
CRD$K_DDB_AUTO_SUPPORT_ENTRY	Literal		82
CRD$K_DDB_BLINK	Literal		82
CRD$K_DDB_FLINK	Literal		82
CRD$K_DDB_LAST_ENTRY	Literal		82
CRD$K_DDB_MEMORY_SIZE	Literal		82
CRD$K_DDB_MENU_DEVICE_NUMB	Literal		82
CRD$K_DDB_MENU_SUPPORT_ENTRY	Literal		82
CRD$K_DDB_MENU_SUPPORT_SIZE	Literal		82
CRD$K_DDB_NUMB_SUPPORT_TABLES	Literal		82
CRD$K_DDB_PTABLE_ENTRY	Literal		82
CRD$K_DDB_STATUS	Literal		82
CRD$K_DDB_SUPBLK_BLOCK_PTR	Literal		82
CRD$K_DEVICE_NAME	Literal		82
CRD$K_DIAGNOSTIC_ERROR	Literal		82
CRD$K_DIAGNOSTIC_NAME	Literal		82
CRD$K_DONE_GENERAL_COMS	Literal		82
CRD$K_DO_NOTHING	Literal		82
CRD$K_DUMMY_10	Literal		82
CRD$K_DUMMY_11	Literal		82
CRD$K_DUMMY_12	Literal		82
CRD$K_DUMMY_13	Literal		82
CRD$K_DUMMY_3	Literal		82
CRD$K_DUMMY_4	Literal		82
CRD$K_DUMMY_5	Literal		82
CRD$K_DUMMY_6	Literal		82
CRD$K_DUMMY_7	Literal		82
CRD$K_DUMMY_8	Literal		82
CRD$K_DUMMY_9	Literal		82
CRD$K_EXIT_CRD	Literal		82
CRD$K_HOOK_POINT	Literal		82
CRD$K_HS_INTERNAL_ERROR	Literal		82
CRD$K_IDC_EVDLB	Literal		82
CRD$K_IDC_EVDLC	Literal		82
CRD$K_IDC_EVDLD	Literal		82
CRD$K_IDC_EVRAD_0	Literal		82
CRD$K_IDC_EVRAD_1	Literal		82
CRD$K_IDC_LAST_DIAGNOSTIC	Literal		82
CRD$K_INTERNAL_ERROR	Literal		82
CRD$K_LAST_ACTION_ROUTINE	Literal		82
CRD$K_LAST_ENTRY	Literal		82
CRD$K_LAST_FUNCTION	Literal		82
CRD$K_LAST_HOOK_POINT	Literal		82

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- -------- ------
CRD$K_LAST_INTERNAL_ERROR Literal     82
CRD$K_LAST_TYPE   Literal     82
CRD$K_LESI_ENWCA  Literal     82
CRD$K_LESI_EVRMB_0  Literal     82
CRD$K_LESI_EVRMB_1  Literal     82
CRD$K_LESI_LAST_DIAGNOSTIC Literal     82
CRD$K_LEVEL_4_PASSED  Literal     82
CRD$K_LOAD_AUTOSIZER  Literal     82
CRD$K_LOAD_ERROR  Literal     82
CRD$K_LOAD_GENERAL_COM  Literal     82
CRD$K_LOAD_LOAD_COM  Literal     82
CRD$K_LOAD_SELECT_COM  Literal     82
CRD$K_LOAD_SET_EVENT_COM Literal     82
CRD$K_LOAD_SET_FLAGS_COM Literal     82
CRD$K_LOAD_START_COM  Literal     82
CRD$K_LOAD_SUP_FILE  Literal     82
CRD$K_MM_INTERNAL_ERROR  Literal     82
CRD$K_NEW_LINE   Literal     82
CRD$K_PREPARATION_ERROR  Literal     82
CRD$K_PREPARATION_ERROR_TEXT Literal     82
CRD$K_RUNTIME   Literal     82
CRD$K_SAME_LINE   Literal     82
CRD$K_SET_EVENT_ARGS  Literal     82
CRD$K_SET_FLAGS_ARGS  Literal     82
CRD$K_SET_UP_DIAGNOSTIC  Literal     82
CRD$K_START   Literal     82
CRD$K_START_AUTOSIZER  Literal     82
CRD$K_START_ERROR  Literal     82
CRD$K_START_QUALIFIERS  Literal     82
CRD$K_STAR_LINE   Literal     82
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal     82
CRD$K_STATE_ADVANCE_GENERAL_COM Literal     82
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal     82
CRD$K_STATE_AUTOSIZER_ERROR Literal     82
CRD$K_STATE_AUTOSIZER_RUNNING Literal     82
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal     82
CRD$K_STATE_DIAGNOSTIC_ERROR Literal     82
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal     82
CRD$K_STATE_DONE_DIAGNOSTICS Literal     82
CRD$K_STATE_DONE_GENERAL_COMS Literal     82
CRD$K_STATE_DUMMY_1  Literal     82
CRD$K_STATE_DUMMY_10  Literal     82
CRD$K_STATE_DUMMY_12  Literal     82
CRD$K_STATE_DUMMY_13  Literal     82
CRD$K_STATE_DUMMY_2  Literal     82
CRD$K_STATE_DUMMY_3  Literal     82
CRD$K_STATE_DUMMY_4  Literal     82
CRD$K_STATE_DUMMY_6  Literal     82
CRD$K_STATE_DUMMY_7  Literal     82
CRD$K_STATE_DUMMY_8  Literal     82
CRD$K_STATE_EXIT_CRD  Literal     82
CRD$K_STATE_INTERNAL_ERROR Literal     82
CRD$K_STATE_LAST_STATE  Literal     82
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_LEVEL_4_PASSED Literal      82
CRD$K_STATE_LOAD_AUTOSIZER Literal      82
CRD$K_STATE_LOAD_ERROR  Literal      82
CRD$K_STATE_LOAD_GENERAL_COM Literal      82
CRD$K_STATE_LOAD_LOAD_COM Literal      82
CRD$K_STATE_LOAD_SELECT_COM Literal      82
CRD$K_STATE_LOAD_SET_EVENT_COM Literal      82
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal      82
CRD$K_STATE_LOAD_START_COM Literal      82
CRD$K_STATE_PREPARATION_ERROR Literal      82
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal      82
CRD$K_STATE_START  Literal      82
CRD$K_STATE_START_AUTOSIZER Literal      82
CRD$K_STATE_START_ERROR  Literal      82
CRD$K_TEST_START_TEXT  Literal      82
CRD$K_TQ_INTERNAL_ERROR  Literal      82
CRD$K_TYPECODE   Literal      82
CRD$K_UDA_0_EVDLB  Literal      82
CRD$K_UDA_0_EVDLC  Literal      82
CRD$K_UDA_0_EVDLD  Literal      82
CRD$K_UDA_0_EVRLA_0  Literal      82
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal      82
CRD$K_UDA_1_EVDLB  Literal      82
CRD$K_UDA_1_EVDLC  Literal      82
CRD$K_UDA_1_EVDLD  Literal      82
CRD$K_UDA_1_EVRLA_0  Literal      82
CRD$K_UDA_1_EVRLA_1  Literal      82
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal      82
CRD$K_UDA_2_EVDLB  Literal      82
CRD$K_UDA_2_EVDLC  Literal      82
CRD$K_UDA_2_EVDLD  Literal      82
CRD$K_UDA_2_EVRLA_0  Literal      82
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal      82
CRD$K_UDA_3_EVDLB  Literal      82
CRD$K_UDA_3_EVDLC  Literal      82
CRD$K_UDA_3_EVDLD  Literal      82
CRD$K_UDA_3_EVRLA_0  Literal      82
CRD$K_UDA_3_EVRLA_1  Literal      82
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal      82
CRD$SET_CTRL_C_FLAG  GlobRout   231 Lib03e
CRD$SET_CTRL_C_HANDLERS  GlobRout   101 Lib03e     66f
DS$GL_FLAGS   Bind        190   192=
    Bind        223   225-
    Bind        256   258=
    Bind        289   291=
    Bind        323   325.
DS$V_CTRLC   Macro    Lib01   258     291     325
DS$V_DISABLCC   Macro    Lib01   192     225
FALSE    Literal  Lib03   192     291
FIRST_HANDLER   RoutForm   101   126.
GET1ST_    Macro    Lib04    82
GET2ND_    Unbound           82
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal      82
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_ACTION_ADVANCE_GENERAL_COM Literal     82
HS$K_ACTION_ADVANCE_STATE Literal     82
HS$K_ACTION_CHANGE_TABLE Literal     82
HS$K_ACTION_DIAGNOSTIC_ERROR Literal     82
HS$K_ACTION_DONE_GENERAL_COMS Literal     82
HS$K_ACTION_DO_NOTHING  Literal     82
HS$K_ACTION_DUMMY_3  Literal     82
HS$K_ACTION_DUMMY_4  Literal     82
HS$K_ACTION_DUMMY_5  Literal     82
HS$K_ACTION_DUMMY_6  Literal     82
HS$K_ACTION_INIT_HS  Literal     82
HS$K_ACTION_INTERNAL_ERROR Literal     82
HS$K_ACTION_LAST_ACTION  Literal     82
HS$K_ACTION_LOAD_ERROR  Literal     82
HS$K_ACTION_LOAD_GENERAL_COM Literal     82
HS$K_ACTION_LOAD_LOAD_COM Literal     82
HS$K_ACTION_LOAD_SELECT_COM Literal     82
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     82
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     82
HS$K_ACTION_LOAD_START_COM Literal     82
HS$K_ACTION_PREPARATION_ERROR Literal     82
HS$K_ACTION_START_ERROR  Literal     82
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     82
HS$K_STATE_ADVANCE_GENERAL_COM Literal     82
HS$K_STATE_CHANGE_TABLE  Literal     82
HS$K_STATE_DIAGNOSTIC_ERROR Literal     82
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     82
HS$K_STATE_DONE_GENERAL_COMS Literal     82
HS$K_STATE_DUMMY_1  Literal     82
HS$K_STATE_DUMMY_2  Literal     82
HS$K_STATE_DUMMY_3  Literal     82
HS$K_STATE_DUMMY_4  Literal     82
HS$K_STATE_DUMMY_6  Literal     82
HS$K_STATE_INIT_HS  Literal     82
HS$K_STATE_INTERNAL_ERROR Literal     82
HS$K_STATE_LAST_STATE  Literal     82
HS$K_STATE_LOAD_ERROR  Literal     82
HS$K_STATE_LOAD_GENERAL_COM Literal     82
HS$K_STATE_LOAD_LOAD_COM Literal     82
HS$K_STATE_LOAD_SELECT_COM Literal     82
HS$K_STATE_LOAD_SET_EVENT_COM Literal     82
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     82
HS$K_STATE_LOAD_START_COM Literal     82
HS$K_STATE_PREPARATION_ERROR Literal     82
HS$K_STATE_START_ERROR  Literal     82
JSB_NO_REGS   Linkage  Lib03    69       72       75       78      134      164      198      231      264      297
JSB_RO_R1   Linkage  Lib03    66      101
MEN$K_HS_STATE_TABLE  Literal     82
MEN$K_MM_STATE_TABLE  Literal     82
MEN$K_TQ_STATE_TABLE  Literal     82
MENU$K_EXIT_AUTOSIZER_ERROR Literal     82
MENU$K_EXIT_INTERNAL_ERROR Literal     82
MENU$K_EXIT_LAST_REASON  Literal     82
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MENU$K_EXIT_OPERATOR_ABORT Literal     82
MENU$K_EXIT_OPERATOR_STOP Literal     82
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     82
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     82
MM$K_ACTION_ADVANCE_STATE Literal     82
MM$K_ACTION_AUTOSIZER_ERROR Literal     82
MM$K_ACTION_BUILD_DDBS  Literal     82
MM$K_ACTION_BUILD_HSTS  Literal     82
MM$K_ACTION_CHANGE_TABLE Literal     82
MM$K_ACTION_DONE_INITS  Literal     82
MM$K_ACTION_DO_NOTHING  Literal     82
MM$K_ACTION_DUMMY_3  Literal     82
MM$K_ACTION_DUMMY_4  Literal     82
MM$K_ACTION_DUMMY_5  Literal     82
MM$K_ACTION_DUMMY_6  Literal     82
MM$K_ACTION_DUMMY_7  Literal     82
MM$K_ACTION_DUMMY_8  Literal     82
MM$K_ACTION_INTERNAL_ERROR Literal     82
MM$K_ACTION_LAST_ACTION  Literal     82
MM$K_ACTION_LOAD_AUTOSIZER Literal     82
MM$K_ACTION_MENU_PROMPT  Literal     82
MM$K_ACTION_PRINT_MENU  Literal     82
MM$K_ACTION_PRINT_MENU_HEADING Literal     82
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     82
MM$K_ACTION_START_AUTOSIZER Literal     82
MM$K_STATE_AUTOSIZER_ERROR Literal     82
MM$K_STATE_AUTOSIZER_RUNNING Literal     82
MM$K_STATE_BUILD_DDBS  Literal     82
MM$K_STATE_BUILD_HSTS  Literal     82
MM$K_STATE_CHANGE_TABLE  Literal     82
MM$K_STATE_DONE_INITS  Literal     82
MM$K_STATE_DUMMY_1  Literal     82
MM$K_STATE_DUMMY_2  Literal     82
MM$K_STATE_DUMMY_3  Literal     82
MM$K_STATE_DUMMY_5  Literal     82
MM$K_STATE_DUMMY_7  Literal     82
MM$K_STATE_DUMMY_8  Literal     82
MM$K_STATE_INTERNAL_ERROR Literal     82
MM$K_STATE_LAST_STATE  Literal     82
MM$K_STATE_LOAD_AUTOSIZER Literal     82
MM$K_STATE_MENU_PROMPT  Literal     82
MM$K_STATE_PRINT_MENU  Literal     82
MM$K_STATE_PRINT_MENU_HEADING Literal     82
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     82
MM$K_STATE_START  Literal     82
MM$K_STATE_START_AUTOSIZER Literal     82
MODNAM    Macro    Lib01    98
NUL2ND_    Macro    Lib04    82
SECOND_HANDLER   RoutForm   101   128.
TQ$K_ACTION_ADVANCE_STATE Literal     82
TQ$K_ACTION_CHANGE_TABLE Literal     82
TQ$K_ACTION_DONE_TEST_QUEUE Literal     82
TQ$K_ACTION_DO_NOTHING  Literal     82
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----- ----------
TQ$K_ACTION_DUMMY_3  Literal     82
TQ$K_ACTION_DUMMY_4  Literal     82
TQ$K_ACTION_DUMMY_5  Literal     82
TQ$K_ACTION_GET_DDB  Literal     82
TQ$K_ACTION_INIT_TQ  Literal     82
TQ$K_ACTION_INTERNAL_ERROR Literal     82
TQ$K_ACTION_LAST_ACTION  Literal     82
TQ$K_STATE_CHANGE_TABLE  Literal     82
TQ$K_STATE_DONE_TEST_QUEUE Literal     82
TQ$K_STATE_DUMMY_1  Literal     82
TQ$K_STATE_DUMMY_2  Literal     82
TQ$K_STATE_DUMMY_3  Literal     82
TQ$K_STATE_DUMMY_4  Literal     82
TQ$K_STATE_DUMMY_5  Literal     82
TQ$K_STATE_GET_DDB  Literal     82
TQ$K_STATE_INIT_TQ  Literal     82
TQ$K_STATE_INTERNAL_ERROR Literal     82
TQ$K_STATE_LAST_STATE  Literal     82
TRUE    Literal  Lib03   225     258


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  -------  ------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDCTRLC.B32;1
    2  Module CRDCTRLC
   53  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   56  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   59  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   62  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  328  Eludom CRDCTRLC


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     @ Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;          -------- Symbols - -    --      Pages Processing
; File         Total    Loaded   Percent      Mapped Time
```

```
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;          671          3          0         46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;          923          0          0         94          00:00.2
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;          486         15          3         62          00:00.2
;  SYS$COMMON:[SYSLIB]LIB.L32;2       18619          3          0       1000          00:04.4




;     COMMAND QUALIFIERS

; BLISS CRDCTRLC.B32/LIST=CRDCTRLC.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDCTRLC.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  74 code + 9 data bytes
; Run Time:    00:19.0
; Elapsed Time:    00:33.3
; Lines/CPU Min:    1037
; Lexemes/CPU-Min: 55658
; Memory Used: 128 pages
; Compilation Complete
```

```
;    0001   0 %TITLE '*** CRDDDB Module'
;    0002   0 MODULE CRDDDB (    ! Build the Device Data Blocks and access them
;    0003   0    IDENT = '01-11'
;    0004   0    ) =
;    0005   0 !++
;    0006   0 ! COPYRIGHT (c) 1980, 1981
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 .
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! Build the Device Data Blocks (dynamically) and define some associated data
;    0032  0 ! structures and accessing routines.
;    0033  0 !
;    0034  0 ! Author:
;    0035  0 !
;    0036  0 ! Jack Stansbury
;    0037  0 !
;    0038  0 ! Creation Date:
;    0039  0 !
;    0040  0 ! May 8, 1982 (Saturday, nearing project's scheduled end-of-coding phase).
;    0041  0 !
;    0042  0 ! Version:
;    0043  0 !
;    0044  0 ! 6.9
;    0045  0 !
;    0046  0 ! Modified By:
;    0047  0 !
;    0048  0 ! 01 Jack Stansbury, Version 1.0, June 21, 1982
;    0049  0 !  Added routine CRD$Dispose (ala Pascal dispose) and changed 'New' routine to
;    0050  0 !  CRD$New and made both routines global. This is primarily for the Menu Test.
;    0051  0 .
;    0052  0 ! 02 Jack Stansbury, Version 1.0, July 16, 1982
;    0053  0 !  Added functionality for Menu Test. Changed the CRD$F_'s to CRD$V_'s. Changed
;    0054  0 !  the error reporting for the New routine.
;    0055  0 !
;    0056  0 ! 03 Jack Stansbury, Version 1.1, July 22, 1982
;    0057  0 !  Changed the CRD$New routine to return the size of the block of allocated memory.
;    0058  0 !  Also changed the DDB's to store the size in the high byte of the Status longword.
;    0059  0 !
;    0060  0 ! 04 Jack Stansbury, Version 1.1, July 27, 1982
;    0061  0 !  Added debugging information to the Dispose and New routines.
;    0062  0 !
;    0063  0 ! 05 Jack Stansbury, Version 1.1, August 20, 1982
;    0064  0 !  Changed the fields in the DDBs, and took the AutoSizer out from the DDBs.
;    0065  0 !
;    0066  0 ! 06 Jack Stansbury, Version 1.1, August 31, 1982
;    0067  0 .  Added a routine to initialize all the status bits in a DDB.
;    0068  0 .
;    0069  0 ! 07 Jack Stansbury, Version 1.1, September 21, 1982
;    0070  0 !  Changed the Init_DDB_Status routine because I changed a lot of the DDB status flags
;    0071  0 !
;    0072  0 ! 08 Jack Stansbury, Version 1.1, September 30, 1982
;    0073  0 !  Took out the Insert_PTable_Ptrs routine.
;    0074  0 !
;    0075  0 . 09 Jack Stansbury, Version 1.1, October 1, 1982
;    0076  0 !  Changed much of the code in the Build_DDB routine. It was wrong.
;    0077  0 !
;    0078  0 . 10 John C'ukaj Version 1.2 8-Apr-1983
```

```
;    0079   0 !  - Build DDB's for LESI/AZTEC and UDA/RA-disk based
;    0080   0 !    systems
;    0081   0 !
;    0082   0 ! 11 Scott Cohen Version 1.4 21-Sep_1983
;    0083   0 !  - Add ENWCA DDB for LESI/AZTEC system
;    0084   0 !
;    0085   0 ! 12 Ron Parker Version 1.5 18-Jun-1984
;    0086   0 !  - Comment out VS125 code
;    0087   0 !
;    0088   0 ! -
```

```
;    0089  0 %SBTTL 'Libraries'
;    0090  1 BEGIN     ! Begin the CRDDDB module
;    0091  1
;    0092  1 LIBRARY
;    0093  1   '$DS';     ! Use Ds.L32 first
;    0094  1
;    0095  1 LIBRARY
;    0096  1   '$Diag';    ! Use Diag.L32 second
;    0097  1
;    0098  1 LIBRARY
;    0099  1   '$CRD';     ! Use CRD.L32 third
;    0100  1
;    0101  1 LIBRARY
;    0102  1   'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0103  1 %SBTTL 'Table of Contents'
;    0104  1
;    0105  1 FORWARD ROUTINE
;    0106  1  CRD$Build_DDBs   : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0107  1      ! Build the queue of DDBs
;    0108  1
;    0109  1  CRD$Dispose   : ADDRESSING_MODE (LONG_RELATIVE),     ![01]
;    0110  1      ! Dispose of a block of non-paged pool memory.    ![01]
;    0111  1
;    0112  1  CRD$Init_DDB_Status  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),    ![06]
;    0113  1      ! Initialize the status bits in a DDB     ![06]
;    0114  1
;    0115  1  CRD$New    : ADDRESSING_MODE (LONG_RELATIVE);     ![01]
;    0116  1      ! Allocate a block of non paged pool memory.    ![01]
```

```
;    0117  1 %SBTTL 'Included Macros and Literals'
;    0118  1
;    0119  1  $DS_DSADef;    ! Define the DSA bits
;    0120  1  $CRD_Literals;    ! Any included macros and literals
```

```
;    0121  1 %SBTTL 'Psect Definitions'
;    0122  1
;    0123  1 PSECT
;    0124  1     PLIT   = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0125  1
;    0126  1 PSECT
;    0127  1     CODE   = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0128  1
;    0129  1 PSECT
;    0130  1     OWN    = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0131  1
;    0132  1 PSECT
;    0133  1     GLOBAL = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0134  1 %SBTTL 'Module-Global Static Storage'
;    0135  1
;    0136  1  ModNam ('CRDDDB');  ! Module name for ErrSup macro
```

```
;    0137  1 %SBTTL 'Global Own Storage'
;    0138  1
;    0139  1 GLOBAL
;    0140  1  CRD$GA_Current_Diagnostic, ! A pointer to the DDB for which we are now running a diagnostic
;    0141  1
;    0142  1  CRD$GA_First_Diagnostic : LONG INITIAL (LONG (0));
;    0143  1       ! A pointer to the first DDB in the queue (i.e., the head).
;    0144  1       ! Some routines
;    0145  1       ! (e.g., CRD$Advance_Diagnostic) depend on this!   [10]
;    0146  1
```

```
;    0147  1 %SBTTL 'CRD$Build_DDBs Routine'
;    0148  1
;    0149  1 GLOBAL ROUTINE CRD$Build_DDBs : NOVALUE =
;    0150  1
;    0151  1 !++
;    0152  1 ! Functional Description:
;    0153  1 !
;    0154  1 ! Build the list of Device_Data_Blocks on the basis of base system type.     [10]
;    0155  1 ! Use the VAX queue instructions to implement
;    0156  1 ! this list. The Data Blocks will be allocated dynamically.
;    0157  1 !
;    0158  1 !
;    0159  1 ! Implicit Inputs:            [10]
;    0160  1 !
;    0161  1 ! The AUTOSIZER has been run and P-Tables exist.
;    0162  1 ! The base system type location (CRD$GB_Base_System) has been
;    0163  1 ! loaded with correct base system code.
;    0164  1 !
;    0165  1 ! Formal Input Parameters:
;    0166  1 !
;    0167  1 ! None
;    0168  1 !
;    0169  1 ! Formal Output Parameters:
;    0170  1 !
;    0171  1 ! None
;    0172  1 !
;    0173  1 ! Side Effects:
;    0174  1 !
;    0175  1 ! None
;    0176  1 !
;    0177  1 ! Completion Codes:
;    0178  1 !
;    0179  1 ! None
;    0180  1 !--
```

```
;    0181  2 BEGIN     ! Routine CRD$Build_DDBs
;    0182  2  MACRO                ![09]
; M  0183  2   Print_DDB_Status (DDB) =          ![09]
; M  0184  2    IF (.CRD$GB_CRD_Debug) THEN        ![09]
; M  0185  2     BEGIN              ![09]
; M  0186  2     LOCAL              ![09]
; M  0187  2      DDB_Ptr : REF Device_Data_Block_Structure;    ![09]
; M  0188  2               ![09]
; M  0189  2     DDB_Ptr = DDB;          ![09]
; M  0190  2     $CRD_Print ($ASCIC ('*** Init_DDB_Status: ',     ![09]
; M  0191  2      'Initializing the !XL DDB status bits to: !XL!/'),   ![09]
; M  0192  2      .DDB_Ptr, .DDB_Ptr [CRD$K_DDB_Status]);     ![09]
; M  0193  2     END              ![09]
;    0194  2  %;                 ![09]
;    0195  2
;    0196  2  MACRO                ![01]
; M  0197  2   Allocate_DDB (Diag_String) =          ![01]
; M  0198  2    Size = CRD$K_DDB_Size;          ![03]
; M  0199  2    IF (NOT CRD$New (Size, DDB_Ptr)) THEN       ![09]
; M  0200  2     BEGIN              ![01]
; M  0201  2     CRD$Internal_Error (CRD$K_Allocation_Error, 0, 0)    ![02]
; M  0202  2     END;               ![09]
; M  0203  2
; M  0204  2    IF .CRD$GB_CRD_Debug THEN         ![09]
; M  0205  2     $CRD_Print ($ASCIC ('  - DDB_Ptr [', Diag_String, '] = !XL(X)!/'), .DDB_Ptr); ![09]
;    0206  2  %;                 ![01]
;    0207  2
;    0208  2  LOCAL                ![03]
;    0209  2   Size,    ! The size of the allocated block     ![03]
;    0210  2   Prev_DDB_Ptr,  ! The previously allocated DDB
;    0211  2   DDB_Ptr    : REF Device_Data_Block_Structure;     ![09]
;    0212  2      ! The allocated DDB.        ![09]
;    0213  2
;    0214  2  BUILTIN              ![09]
;    0215  2   INSQUE;    ! Use the VAX InsQue machine instruction.    ![09]
;    0216  2
;    0217  2
```

```
;    0218  2 !+                [10]
;    0219  2 ! Build DDB's on the basis of base system type
;    0220  2 !-
;    0221  2 SELECTONE .CRD$GB_Base_System OF
;    0222  2     SET
;    0223  2     [CRD$K_Base_System_IDC]:
;    0224  3  BEGIN
;    0225  3  !+
;    0226  3  ! IDC based System
;    0227  3  ! Allocate a Device Data Block for EVRAD 0 (for the R80 or the RL02 test):
;    0228  3  !-
;    0229  3
;    0230  3  Allocate_DDB ('EVRAD_0');
;    0231  3
;    0232  3  DDB_Ptr [CRD$K_DDB_FLink] = .DDB_Ptr;          ![09]
;    0233  3  DDB_Ptr [CRD$K_DDB_BLink] = .DDB_Ptr;          ![09]
;    0234  3
;    0235  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL IDC_Hdwr_Sprt_Table [CRD$K_IDC EVRAD_0, 0];
;    0236  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0237  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0238  3
;    0239  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0240  3
;    0241  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device> = True;     ![09]
;    0242  3      ! This is an essential device - it must be tested
;    0243  3
;    0244  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Check_Device_Type> - True;     ![09]
;    0245  3      ! Check the Device_Type field in the Test_Start_Text when outputting
;    0246  3      ! the text. This is because this diagnostic will test either the R80 or
;    0247  3      ! the RL02 on DQA0, depending on what is there.
;    0248  3
;    0249  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0250  3      ! Store the actual size of the DDB.     ![03]
;    0251  3
;    0252  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0253  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0254  3
;    0255  3  CRD$GA_First_Diagnostic   = .DDB_Ptr;          ![09]
;    0256  3      ! This will point to the EVRAD_0 DDB. It will be the first one.  ![05]
;    0257  3
;    0258  3  CRD$GA_Current_Diagnostic = .DDB_Ptr;          ![09]
;    0259  3      ! This will be the current DDB also.     ![05]
;    0260  3
;    0261  3  !+
;    0262  3  ! Allocate a Device Data Block for EVRAD_1 (for the RL02 test):
;    0263  3  !-
;    0264  3
;    0265  3  Allocate_DDB ('EVRAD_1');
;    0266  3
;    0267  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0268  3
;    0269  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_IDC Hdwr_Sprt_Table [CRD$K_IDC_EVRAD_1, 0];
;    0270  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0271  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0272  3
```

```
;    0273  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0274  3
;    0275  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device> = True;     ![09]
;    0276  3      ! This is an essential device - it must be tested
;    0277  3
;    0278  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0279  3      ! Store the actual size of the DDB.     ![03]
;    0280  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0281  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0282  3
;    0283  3  !+
;    0284  3  ! Allocate a Device Data Block for EVDLB (for the DMF32S device - the Combo sync port):
;    0285  3  !-
;    0286  3
;    0287  3  Allocate_DDB ('EVDLB   ');
;    0288  3
;    0289  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);         ![09]
;    0290  3
;    0291  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_IDC_Hdwr_Sprt_Table [CRD$K_IDC_EVDLB, 0];
;    0292  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0293  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0294  3
;    0295  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0296  3
;    0297  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0298  3      ! Store the actual size of the DDB.     ![03]
;    0299  3
;    0300  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0301  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0302  3
;    0303  3  !+
;    0304  3  ! Allocate a Device Data Block for EVDLC (for the DMF32A device - the Combo async port):
;    0305  3  !-
;    0306  3
;    0307  3  Allocate_DDB ('EVDLC   ');
;    0308  3
;    0309  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);         ![09]
;    0310  3
;    0311  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_IDC Hdwr_Sprt_Table [CRD$K_IDC_EVDLC, 0];
;    0312  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0313  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0314  3
;    0315  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0316  3
;    0317  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential Device> = True;     ![09]
;    0318  3      ! This is an essential device - it must be tested
;    0319  3
;    0320  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> - .Size;      ![09]
;    0321  3      ! Store the actual size of the DDB.     ![03]
;    0322  3
;    0323  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0324  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0325  3
;    0326  3  !+
;    0327  3  ! Allocate a Device Data Block for EVDLD (for the DMF32P device - the Combo paralle  port):
```

```
;    0328  3  !-
;    0329  3
;    0330  3  Allocate_DDB ('EVDLD  ');
;    0331  3
;    0332  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0333  3
;    0334  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_IDC_Hdwr_Sprt_Table [CRD$K_IDC_EVDLD, 0];
;    0335  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0336  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;         ![09]
;    0337  3
;    0338  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status b'ts      ![09]
;    0339  3
;    0340  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> - .S'ze;      ![09]
;    0341  3      ! Store the actual size of the DDB.     ![03]
;    0342  3
;    0343  3  Print_DDB_Status (.DDB_Ptr);           ![09]
;    0344  3  RETURN;
;    0345  2  END;
;    0346  2
;    0347  2
```

```
;    0348  2
;    0349  2      !+
;    0350  2      ! LESI/AZTEC disk based system
;    0351  2      !-
;    0352  2      [CRD$K_Base_System_LESI]:
;    0353  3  BEGIN
;    0354  3  !+
;    0355  3  ! Allocate a Device Data Block for EVRMB_0 (for testing Drive 0)
;    0356  3  !-
;    0357  3
;    0358  3  Allocate_DDB ('EVRMB_0');
;    0359  3
;    0360  3  DDB_Ptr [CRD$K_DDB_FLink] = .DDB_Ptr;          ![09]
;    0361  3  DDB_Ptr [CRD$K_DDB_BLink] = .DDB_Ptr;          ![09]
;    0362  3
;    0363  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_LESI_Hdwr_Sprt_Table [CRD$K_LESI_EVRMB_0, 0];
;    0364  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0365  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0366  3
;    0367  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0368  3
;    0369  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device> = True;     ![09]
;    0370  3      ! This is an essential device - it must be tested
;    0371  3
;    0372  3
;    0373  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0374  3      ! Store the actual size of the DDB.     ![03]
;    0375  3
;    0376  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0377  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0378  3
;    0379  3  CRD$GA_First_Diagnostic   = .DDB_Ptr;          ![09]
;    0380  3      ! This will point to the EVRMB_0 DDB. It will be the first one.  ![05]
;    0381  3
;    0382  3  CRD$GA_Current_Diagnostic = .DDB_Ptr;          ![09]
;    0383  3      ! This will be the current DDB also.     ![05]
;    0384  3
;    0385  3  !+
;    0386  3  ! Allocate a Device Data Block for EVRMB_1 (Drive 1)
;    0387  3  !-
;    0388  3
;    0389  3  Allocate_DDB ('EVRMB_1');
;    0390  3
;    0391  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0392  3
;    0393  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_LESI_Hdwr_Sprt_Table [CRD$K_LESI_EVRMB_1, 0];
;    0394  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0395  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0396  3
;    0397  3  CRD$Init_DDB Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0398  3
;    0399  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential Device> = True;     ![09]
;    0400  3      ! This is an essential device - it must be tested
;    0401  3
;    0402  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
```

```
;    0403  3      ! Store the actual size of the DDB.      ![03]
;    0404  3   Print_DDB_Status (.DDB_Ptr);          ![09]
;    0405  3   Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0406  3
;    0407  3
;    0408  3
;    0409  3 ! BEGIN              ![11]
;    0410  3 !  BIND VS125_Device_Name = $ASCIC ('VS125');       ![11]
;    0411  3 !  REGISTER VS125_PTable;          ![11]
;    0412  3 !
;    0413  3 !  VS125_PTable = CRD$Find_PTable (VS125_Device_Name);      ![11]
;    0414  3 !
;    0415  3 !  IF (.VS125_PTable NEQ 0)         ![11]
;    0416  3 !  THEN             ![11]
;    0417  3 !     BEGIN           ![11]
;    0418  3 !   !+            ![11]
;    0419  3 !   ! Allocate a Device Data Block for ENWCA      ![11]
;    0420  3 !   !-            ![11]
;    0421  3 !
;    0422  3 !   Allocate_DDB ('ENWCA');         ![11]
;    0423  3 !
;    0424  3 !   INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);       ![11]
;    0425  3 !
;    0426  3 !   DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_LESI_Hdwr_Sprt_Table [CRD$K_LESI_ENWCA, 0]; ![11]
;    0427  3 !   DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;      ![11]
;    0428  3 !   DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;      ![11]
;    0429  3 !
;    0430  3 !   CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits    ![11]
;    0431  3 !
;    0432  3 !   DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device> = True;   ![11]
;    0433  3 !       ! This is an essential device - it must be tested ![11]
;    0434  3 !
;    0435  3 !   DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;    ![11]
;    0436  3 !     ! Store the actual size of the DDB.
;    0437  3 !   Print_DDB_Status (.DDB_Ptr);        ![11]
;    0438  3 !   Prev_DDB_Ptr = .DDB_Ptr;        ![11]
;    0439  3 !     END;            ![11]
;    0440  3 ! END;              ![11]
;    0441  3
;    0442  2  END;
```

```
;    0443  2      !+
;    0444  2      ! UDA based system with no RA60 Drive 1, RA80 or 81 Drive 0
;    0445  2      !-
;    0446  2      [CRD$K_Base_System_UDA_0]:
;    0447  3  BEGIN
;    0448  3  !+
;    0449  3  ! Allocate a Device Data Block for EVRLA_0 (Drive 0):
;    0450  3  !-
;    0451  3
;    0452  3  Allocate_DDB ('EVRLA_0');
;    0453  3
;    0454  3  DDB_Ptr [CRD$K_DDB_FLink] = .DDB_Ptr;          ![09]
;    0455  3  DDB_Ptr [CRD$K_DDB_BLink] = .DDB_Ptr;          ![09]
;    0456  3
;    0457  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_0_Hdwr_Sprt_Table [CRD$K_UDA_0_EVRLA_0, 0];
;    0458  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0459  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0460  3
;    0461  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0462  3
;    0463  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device> = True;     ![09]
;    0464  3      ! This is an essential device - it must be tested
;    0465  3
;    0466  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Check_Device_Type> = True;     ![09]
;    0467  3      ! Check the Device_Type field in the Test_Start_Text when outputting
;    0468  3      ! the text. This is because this diagnostic will test either the R80 or
;    0469  3      ! the RL02 on DQA0, depending on what is there.
;    0470  3
;    0471  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0472  3      ! Store the actual size of the DDB.     ![03]
;    0473  3
;    0474  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0475  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0476  3
;    0477  3  CRD$GA_First_Diagnostic   = .DDB_Ptr;          ![09]
;    0478  3      ! This will point to the EVRLA_0 DDB. It will be the first one.  ![05]
;    0479  3
;    0480  3  CRD$GA_Current_Diagnostic = .DDB_Ptr;          ![09]
;    0481  3      ! Th's will be the current DDB also.     ![05]
;    0482  3
;    0483  3
;    0484  3  !+
;    0485  3  ! Allocate a Device Data Block for EVDLB (for the DMF32S device - the Combo sync port):
;    0486  3  !-
;    0487  3
;    0488  3  Allocate_DDB ('EVDLB   ');
;    0489  3
;    0490  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0491  3
;    0492  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_0_Hdwr_Sprt_Tab e [CRD$K_UDA 0 EVDLB, 0];
;    0493  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0494  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;         ![09]
;    0495  3
;    0496  3  CRD$Init_DDB_Status (.DDB_Ptr); ! In'tialize the status b'ts      .[09]
;    0497  3
```

```
;    0498  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0499  3      ! Store the actual size of the DDB.     ![03]
;    0500  3
;    0501  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0502  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0503  3
;    0504  3  !+
;    0505  3  ! Allocate a Device Data Block for EVDLC (for the DMF32A device - the Combo async port):
;    0506  3  !-
;    0507  3
;    0508  3  Allocate_DDB ('EVDLC   ');
;    0509  3
;    0510  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0511  3
;    0512  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_0_Hdwr_Sprt_Table [CRD$K_UDA_0_EVDLC, 0];
;    0513  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0514  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0515  3
;    0516  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0517  3
;    0518  3
;    0519  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0520  3      ! Store the actual size of the DDB.     ![03]
;    0521  3
;    0522  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0523  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0524  3
;    0525  3  !+
;    0526  3  ! Allocate a Device Data Block for EVDLD (for the DMF32P device - the Combo parallel port):
;    0527  3  !-
;    0528  3
;    0529  3  Allocate_DDB ('EVDLD   ');
;    0530  3
;    0531  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0532  3
;    0533  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_0_Hdwr_Sprt_Table [CRD$K_UDA_0_EVDLD, 0];
;    0534  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0535  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0536  3
;    0537  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0538  3
;    0539  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0540  3      ! Store the actual size of the DDB.     ![03]
;    0541  3
;    0542  3  Print_DDB Status (.DDB Ptr);          ![09]
;    0543  3  RETURN;
;    0544  2  END;
;    0545  2
;    0546  2
```

```
;    0547  2      !+
;    0548  2      ! UDA based system with RA60 drive 1, RA80 or 81 Drive 0
;    0549  2      !-
;    0550  2      [CRD$K_Base_System_UDA_1]:
;    0551  3  BEGIN
;    0552  3  !+
;    0553  3  ! Allocate a Device Data Block for EVRLA_0 (for Drive 0):
;    0554  3  !-
;    0555  3
;    0556  3  Allocate_DDB ('EVRLA_0');
;    0557  3
;    0558  3  DDB_Ptr [CRD$K_DDB_FLink] = .DDB_Ptr;          ![09]
;    0559  3  DDB_Ptr [CRD$K_DDB_BLink] = .DDB_Ptr;          ![09]
;    0560  3
;    0561  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_1_Hdwr Sprt_Table [CRD$K_UDA_1_EVRLA_0, 0];
;    0562  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0563  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0564  3
;    0565  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0566  3
;    0567  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device> = True;     ![09]
;    0568  3      ! This is an essential device - it must be tested
;    0569  3
;    0570  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Check_Device_Type> = True;     ![09]
;    0571  3      ! Check the Device_Type field in the Test_Start_Text when outputting
;    0572  3      ! the text. This is because this diagnostic will test either the R80 or
;    0573  3      ! the RL02 on DQA0, depending on what is there.
;    0574  3
;    0575  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0576  3      ! Store the actual size of the DDB.     ![03]
;    0577  3
;    0578  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0579  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0580  3
;    0581  3  CRD$GA_First_Diagnostic   = .DDB_Ptr;          ![09]
;    0582  3      ! This will point to the EVRLA_0 DDB. It will be the first one.  ![05]
;    0583  3
;    0584  3  CRD$GA_Current_Diagnostic = .DDB_Ptr;          ![09]
;    0585  3      ! This will be the current DDB also.     ![05]
;    0586  3
;    0587  3  !+
;    0588  3  ! Allocate a Device Data Block for EVRLA_1 (for Drive 1):
;    0589  3  !-
;    0590  3
;    0591  3  Allocate_DDB ('EVRLA_1');
;    0592  3
;    0593  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0594  3
;    0595  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_1_Hdwr_Sprt_Table [CRD$K_UDA 1_EVRLA_1, 0];
;    0596  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0597  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0598  3
;    0599  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0600  3
;    0601  3  DDB Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device> = True;     ![09]
```

```
;    0602  3      ! This is an essential device - it must be tested
;    0603  3
;    0604  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0605  3      ! Store the actual size of the DDB.     ![03]
;    0606  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0607  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0608  3
;    0609  3  !+
;    0610  3  ! Allocate a Device Data Block for EVDLB (for the DMF32S device - the Combo sync port):
;    0611  3  !-
;    0612  3
;    0613  3  Allocate_DDB ('EVDLB   ');
;    0614  3
;    0615  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0616  3
;    0617  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_1_Hdwr_Sprt_Table [CRD$K_UDA_1_EVDLB, 0];
;    0618  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0619  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0620  3
;    0621  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0622  3
;    0623  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0624  3      ! Store the actual size of the DDB.     ![03]
;    0625  3
;    0626  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0627  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0628  3
;    0629  3  !+
;    0630  3  ! Allocate a Device Data Block for EVDLC (for the DMF32A device - the Combo async port):
;    0631  3  !-
;    0632  3
;    0633  3  Allocate_DDB ('EVDLC   ');
;    0634  3
;    0635  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0636  3
;    0637  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_1_Hdwr_Sprt_Table [CRD$K_UDA_1_EVDLC, 0];
;    0638  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0639  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0640  3
;    0641  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0642  3
;    0643  3
;    0644  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0645  3      ! Store the actual size of the DDB.     ![03]
;    0646  3
;    0647  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0648  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0649  3
;    0650  3  !+
;    0651  3  ! Allocate a Device Data Block for EVDLD (for the DMF32P device - the Combo parallel port):
;    0652  3  !-
;    0653  3
;    0654  3  Allocate_DDB ('EVDLD   ');
;    0655  3
;    0656  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
```

```
;    0657  3
;    0658  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_1_Hdwr_Sprt_Table [CRD$K_UDA_1_EVDLD, 0];
;    0659  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0660  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0661  3
;    0662  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0663  3
;    0664  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0665  3      ! Store the actual size of the DDB.     ![03]
;    0666  3
;    0667  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0668  3  RETURN;
;    0669  2  END;
;    0670  2
;    0671  2
;    0672  2
```

```
;    0673  2
;    0674  2
;    0675  2
;    0676  2     !+
;    0677  2     ! UDA based system with no RA60 Drive 1, RA60 Drive 0
;    0678  2     !-
;    0679  2     [CRD$K_Base_System_UDA_2]:
;    0680  3  BEGIN
;    0681  3  !+
;    0682  3  ! Allocate a Device Data Block for EVRLA_0 (Drive 0):
;    0683  3  !-
;    0684  3
;    0685  3  Allocate_DDB ('EVRLA_0');
;    0686  3
;    0687  3  DDB_Ptr [CRD$K_DDB_FLink] = .DDB_Ptr;          ![09]
;    0688  3  DDB_Ptr [CRD$K_DDB_BLink] = .DDB_Ptr;          ![09]
;    0689  3
;    0690  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_2_Hdwr_Sprt_Tab e [CRD$K UDA_2_EVRLA_0, 0];
;    0691  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0692  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0693  3
;    0694  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0695  3
;    0696  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Dev'ce> = True;     .[09]
;    0697  3      ! This is an essential device - it must be tested
;    0698  3
;    0699  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Check_Device_Type> = True;     ![09]
;    0700  3      ! Check the Device_Type field in the Test_Start_Text when outputting
;    0701  3      ! the text. This is because this diagnostic will test either the R80 or
;    0702  3      ! the RL02 on DQA0, depending on what is there.
;    0703  3
;    0704  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0705  3      ! Store the actual size of the DDB.     ![03]
;    0706  3
;    0707  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0708  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0709  3
;    0710  3  CRD$GA_First_Diagnostic   = .DDB_Ptr;          ![09]
;    0711  3      ! This will point to the EVRLA_0 DDB. It will be the first one.  ![05]
;    0712  3
;    0713  3  CRD$GA_Current_Diagnostic = .DDB_Ptr;          ![09]
;    0714  3      ! This will be the current DDB also.     ![05]
;    0715  3
;    0716  3
;    0717  3  !+
;    0718  3  ! Allocate a Device Data Block for EVDLB (for the DMF32S device - the Combo sync port):
;    0719  3  !-
;    0720  3
;    0721  3  Allocate_DDB ('EVDLB   ');
;    0722  3
;    0723  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0724  3
;    0725  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA 2 Hdwr_Sprt_Table [CRD$K_UDA_2_EVDLB, 0];
;    0726  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0727  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;          [09]
```

```
;    0728  3
;    0729  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0730  3
;    0731  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0732  3      ! Store the actual size of the DDB.     ![03]
;    0733  3
;    0734  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0735  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0736  3
;    0737  3  !+
;    0738  3  ! Allocate a Device Data Block for EVDLC (for the DMF32A device - the Combo async port):
;    0739  3  !-
;    0740  3
;    0741  3  Allocate_DDB ('EVDLC   ');
;    0742  3
;    0743  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0744  3
;    0745  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_2_Hdwr Sprt_Table [CRD$K_UDA_2_EVDLC, 0];
;    0746  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0747  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0748  3
;    0749  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0750  3
;    0751  3
;    0752  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0753  3      ! Store the actual size of the DDB.     ![03]
;    0754  3
;    0755  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0756  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0757  3
;    0758  3  !+
;    0759  3  ! Allocate a Device Data Block for EVDLD (for the DMF32P device - the Combo parallel port):
;    0760  3  !-
;    0761  3
;    0762  3  Allocate_DDB ('EVDLD   ');
;    0763  3
;    0764  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0765  3
;    0766  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_2_Hdwr_Sprt_Table [CRD$K_UDA_2_EVDLD, 0];
;    0767  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0768  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0769  3
;    0770  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0771  3
;    0772  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0773  3      ! Store the actual size of the DDB.     ![03]
;    0774  3
;    0775  3  Print_DDB_Status (.DDB Ptr);          ![09]
;    0776  3  RETURN;
;    0777  2  END;
;    0778  2
;    0779  2
```

```
;    0780  2      !+
;    0781  2      ! UDA based system with RA60 drive 1, RA60 Drive 0
;    0782  2      !-
;    0783  2      [CRD$K_Base_System_UDA_3]:
;    0784  3  BEGIN
;    0785  3  !+
;    0786  3  ! Allocate a Device Data Block for EVRLA_0 (for Drive 0):
;    0787  3  !-
;    0788  3
;    0789  3  Allocate_DDB ('EVRLA_0');
;    0790  3
;    0791  3  DDB_Ptr [CRD$K_DDB_FLink] = .DDB_Ptr;          ![09]
;    0792  3  DDB_Ptr [CRD$K_DDB_BLink] = .DDB_Ptr;          ![09]
;    0793  3
;    0794  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_3_Hdwr_Sprt_Table [CRD$K_UDA 3_EVRLA_0, 0];
;    0795  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0796  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0797  3
;    0798  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0799  3
;    0800  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device> = True;     ![09]
;    0801  3      ! This is an essential device - it must be tested
;    0802  3
;    0803  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Check_Device_Type> - True;     ![09]
;    0804  3      ! Check the Device_Type field in the Test_Start_Text when outputting
;    0805  3      ! the text. This is because this diagnostic will test either the R80 or
;    0806  3      ! the RL02 on DQA0, depending on what is there.
;    0807  3
;    0808  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0809  3      ! Store the actual size of the DDB.     ![03]
;    0810  3
;    0811  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0812  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0813  3
;    0814  3  CRD$GA_First_Diagnostic   = .DDB_Ptr;          ![09]
;    0815  3      ! This will point to the EVRLA_0 DDB. It will be the first one.  ![05]
;    0816  3
;    0817  3  CRD$GA_Current_Diagnostic = .DDB_Ptr;          ![09]
;    0818  3      ! This will be the current DDB also.     ![05]
;    0819  3
;    0820  3  !+
;    0821  3  ! Allocate a Device Data Block for EVRLA_1 (for Drive 1):
;    0822  3  !-
;    0823  3
;    0824  3  Allocate_DDB ('EVRLA_1');
;    0825  3
;    0826  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0827  3
;    0828  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_3_Hdwr_Sprt_Table [CRD$K_UDA_3_EVRLA_1, 0];
;    0829  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0830  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;
;    0831  3
;    0832  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0833  3
;    0834  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device> = True;     ![09]
```

```
;    0835  3      ! This is an essential device - it must be tested
;    0836  3
;    0837  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0838  3      ! Store the actual size of the DDB.     ![03]
;    0839  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0840  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0841  3
;    0842  3  !+
;    0843  3  ! Allocate a Device Data Block for EVDLB (for the DMF32S device - the Combo sync port):
;    0844  3  !-
;    0845  3
;    0846  3  Allocate_DDB ('EVDLB   ');
;    0847  3
;    0848  3  INSQUE (.DDB_Ptr, .Prev_DDB Ptr);          ![09]
;    0849  3
;    0850  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL UDA_3_Hdwr_Sprt_Table [CRD$K_UDA_3_EVDLB, 0];
;    0851  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0852  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0853  3
;    0854  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0855  3
;    0856  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0857  3      ! Store the actual size of the DDB.     ![03]
;    0858  3
;    0859  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0860  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0861  3
;    0862  3  !+
;    0863  3  ! Allocate a Device Data Block for EVDLC (for the DMF32A device - the Combo async port):
;    0864  3  !-
;    0865  3
;    0866  3  Allocate_DDB ('EVDLC   ');
;    0867  3
;    0868  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
;    0869  3
;    0870  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_3_Hdwr_Sprt_Table [CRD$K_UDA_3_EVDLC, 0];
;    0871  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0872  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       - 0;        ![09]
;    0873  3
;    0874  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0875  3
;    0876  3
;    0877  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> - .Size;      ![09]
;    0878  3      ! Store the actual size of the DDB.     ![03]
;    0879  3
;    0880  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0881  3  Prev_DDB_Ptr = .DDB_Ptr;          ![09]
;    0882  3
;    0883  3  !+
;    0884  3  ! Allocate a Device Data Block for EVDLD (for the DMF32P device   the Combo parallel port):
;    0885  3  !-
;    0886  3
;    0887  3  Allocate_DDB ('EVDLD   ');
;    0888  3
;    0889  3  INSQUE (.DDB_Ptr, .Prev_DDB_Ptr);          ![09]
```

```
;    0890  3
;    0891  3  DDB_Ptr [CRD$K_DDB_Auto_Support_Entry] = CRD$AL_UDA_3_Hdwr_Sprt_Table [CRD$K_UDA_3_EVDLD, 0];
;    0892  3  DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = 0;
;    0893  3  DDB_Ptr [CRD$K_DDB_PTable_Entry]       = 0;        ![09]
;    0894  3
;    0895  3  CRD$Init_DDB_Status (.DDB_Ptr); ! Initialize the status bits      ![09]
;    0896  3
;    0897  3  DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size> = .Size;      ![09]
;    0898  3      ! Store the actual size of the DDB.     ![03]
;    0899  3
;    0900  3  Print_DDB_Status (.DDB_Ptr);          ![09]
;    0901  3  RETURN;
;    0902  2  END;
;    0903  2
;    0904  2
;    0905  2
```

```
;   0906  2
;   0907  2     TES;
;   0908  2
;   0909  2
;   0910  1 END;     ! Routine CRD$Build_DDBs


          .TITLE  CRDDDB *** CRDDDB Module
          .IDENT  \01-11\

          .PSECT  WORK,NOEXE,2

          00000 CRD$GA_CURRENT_DIAGNOSTIC::
          .BLKB   4
       00000000  00004 CRD$GA_FIRST_DIAGNOSTIC::
          .LONG   0           ;

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

    42  44  44  44  52  43  06  00000 P.AAA:  .ASCII  <6>\CRDDDB\          ;
45  5B  20  72  74  50  5F  42  44  44  20  2D  20  20  20  00007 P.AAB:  .ASCII  \   - DDB_Ptr [EVRAD_0] = !XL(X)!/\       ;
58  28  4C  58  21  20  3D  20  5D  30  5F  44  41  52  56  00016             ;
      2F  21  29  00025             ;
53  5F  42  44  44  5F  74  69  6E  49  20  2A  2A  2A  43  00028 P.AAC:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69  6C  61  69  74  69  6E  49  20  3A  73  75  74  61  74  00037             ;
        21  20  65  68  74  20  67  6E  69  7A  00046             ;
62  20  73  75  74  61  74  73  20  42  44  44  20  4C  58  00050    .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F  21  4C  58  21  20  3A  6F  74  20  73  74  69  0005F           ;
45  5B  20  72  74  50  5F  42  44  44  20  2D  20  20  20  0006C P.AAD:  .ASCII  \   - DDB_Ptr [EVRAD_1] = !XL(X)!/\       ;
58  28  4C  58  21  20  3D  20  5D  31  5F  44  41  52  56  0007B             ;
      2F  21  29  0008A             ;
53  5F  42  44  44  5F  74  69  6E  49  20  2A  2A  2A  43  0008D P.AAE:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69  6C  61  69  74  69  6E  49  20  3A  73  75  74  61  74  0009C             ;
        21  20  65  68  74  20  67  6E  69  7A  000AB             ;
62  20  73  75  74  61  74  73  20  42  44  44  20  4C  58  000B5    .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F  21  4C  58  21  20  3A  6F  74  20  73  74  69  000C4           ;
45  5B  20  72  74  50  5F  42  44  44  20  2D  20  20  20  000D1 P.AAF:  .ASCII  \   - DDB_Ptr [EVDLB  ] = !XL(X)!/\       ;
58  28  4C  58  21  20  3D  20  5D  20  20  42  4C  44  56  000E0             ;
      2F  21  29  000EF             ;
53  5F  42  44  44  5F  74  69  6E  49  20  2A  2A  2A  43  000F2 P.AAG:  .ASCII  \C*** Init_DDB_Status: Initializing the .\  ;
69  6C  61  69  74  69  6E  49  20  3A  73  75  74  61  74  00101             ;
        21  20  65  68  74  20  67  6E  69  7A  00110             ;
62  20  73  75  74  61  74  73  20  42  44  44  20  4C  58  0011A    .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F  21  4C  58  21  20  3A  6F  74  20  73  74  69  00129           ;
45  5B  20  72  74  50  5F  42  44  44  20  2D  20  20  20  00136 P.AAH:  .ASCII  \   - DDB_Ptr [EVDLC  ] = !XL(X)!/\       ;
58  28  4C  58  21  20  3D  20  5D  20  20  43  4C  44  56  00145             ;
      2F  21  29  00154             ;
53  5F  42  44  44  5F  74  69  6E  49  20  2A  2A  2A  43  00157 P.AAI:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69  6C  61  69  74  69  6E  49  20  3A  73  75  74  61  74  00166             ;
        21  20  65  68  74  20  67  6E  69  7A  00175             ;
62  20  73  75  74  61  74  73  20  42  44  44  20  4C  58  0017F    .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F  21  4C  58  21  20  3A  6F  74  20  73  74  69  0018E           ;
45  5B  20  72  74  50  5F  42  44  44  20  2D  20  20  20  0019B P.AAJ:  .ASCII  \   - DDB_Ptr [EVDLD  ] = !XL(X)!/\       ;
58  28  4C  58  21  20  3D  20  5D  20  20  44  4C  44  56  001AA             ;
      2F  21  29  001B9             ;
```

```
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  001BC P.AAK:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  001CB              ;
      21 20 65 68 74 20 67 6E 69 7A  001DA                      ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  001E4     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69  001F3                 ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  00200 P.AAL:  .ASCII  \   - DDB_Ptr [EVRMB_0] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 30 5F 42 4D 52 56  0020F              ;
      2F 21 29  0021E                         ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  00221 P.AAM:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  00230              ;
      21 20 65 68 74 20 67 6E 69 7A  0023F                      ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  00249     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69  00258                 ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  00265 P.AAN:  .ASCII  \   - DDB_Ptr [EVRMB_1] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 31 5F 42 4D 52 56  00274              ;
      2F 21 29  00283                         ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  00286 P.AAO:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  00295              ;
      21 20 65 68 74 20 67 6E 69 7A  002A4                      ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  002AE     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69  002BD                 ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  002CA P.AAP:  .ASCII  \   - DDB_Ptr [EVRLA_0] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 30 5F 41 4C 52 56  002D9              ;
      2F 21 29  002E8                         ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  002EB P.AAQ:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  002FA              ;
      21 20 65 68 74 20 67 6E 69 7A  00309                      ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  00313     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69  00322                 ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  0032F P.AAR:  .ASCII  \   - DDB_Ptr [EVDLB  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 42 4C 44 56  0033E              ;
      2F 21 29  0034D                         ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  00350 P.AAS:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  0035F              ;
      21 20 65 68 74 20 67 6E 69 7A  0036E                      ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  00378     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69  00387                 ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  00394 P.AAT:  .ASCII  \   - DDB_Ptr [EVDLC  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 43 4C 44 56  003A3              ;
      2F 21 29  003B2                         ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  003B5 P.AAU:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  003C4              ;
      21 20 65 68 74 20 67 6E 69 7A  003D3                      ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  003DD     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69  003EC                 ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  003F9 P.AAV:  .ASCII  \   - DDB_Ptr [EVDLD  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 44 4C 44 56  00408              ;
      2F 21 29  00417                         ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  0041A P.AAW:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  00429              ;
      21 20 65 68 74 20 67 6E 69 7A  00438                      ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  00442     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69  00451                 ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  0045E P.AAX:  .ASCII  \   - DDB_Ptr [EVRLA_0] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 30 5F 41 4C 52 56  0046D              ;
```

```
                2F 21 29 0047C                  ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  0047F P.AAY:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  0048E                ;
      21 20 65 68 74 20 67 6E 69 7A 0049D                           ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  004A7     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 004B6                       ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  004C3 P.AAZ:  .ASCII  \   - DDB_Ptr [EVRLA_1] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 31 5F 41 4C 52 56  004D2                ;
                2F 21 29 004E1                  ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  004E4 P.ABA:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  004F3                ;
      21 20 65 68 74 20 67 6E 69 7A 00502                           ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  0050C     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 0051B                       ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  00528 P.ABB:  .ASCII  \   - DDB_Ptr [EVDLB  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 42 4C 44 56  00537                ;
                2F 21 29 00546                  ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  00549 P.ABC:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  00558                ;
      21 20 65 68 74 20 67 6E 69 7A 00567                           ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  00571     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 00580                       ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  0058D P.ABD:  .ASCII  \   - DDB_Ptr [EVDLC  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 43 4C 44 56  0059C                ;
                2F 21 29 005AB                  ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  005AE P.ABE:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  005BD                ;
      21 20 65 68 74 20 67 6E 69 7A 005CC                           ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  005D6     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 005E5                       ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  005F2 P.ABF:  .ASCII  \   - DDB_Ptr [EVDLD  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 44 4C 44 56  00601                ;
                2F 21 29 00610                  ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  00613 P.ABG:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  00622                ;
      21 20 65 68 74 20 67 6E 69 7A 00631                           ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  0063B     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 0064A                       ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  00657 P.ABH:  .ASCII  \   - DDB_Ptr [EVRLA_0] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 30 5F 41 4C 52 56  00666                ;
                2F 21 29 00675                  ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  00678 P.ABI:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  00687                ;
      21 20 65 68 74 20 67 6E 69 7A 00696                           ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  006A0     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 006AF                       ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  006BC P.ABJ:  .ASCII  \   - DDB_Ptr [EVDLB  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 42 4C 44 56  006CB                ;
                2F 21 29 006DA                  ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43  006DD P.ABK:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74  006EC                ;
      21 20 65 68 74 20 67 6E 69 7A 006FB                           ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58  00705     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 00714                       ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20  00721 P.ABL:  .ASCII  \   - DDB_Ptr [EVDLC  ] = !XL(X)!/\       ;
```

```
58 28 4C 58 21 20 3D 20 5D 20 20 43 4C 44 56 00730                ;
      2F 21 29 0073F                ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43 00742 P.ABM:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74 00751                ;
      21 20 65 68 74 20 67 6E 69 7A 00760                ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58 0076A     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 00779                ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20 00786 P.ABN:  .ASCII  \   - DDB_Ptr [EVDLD  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 44 4C 44 56 00795                ;
      2F 21 29 007A4                ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43 007A7 P.ABO:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74 007B6                ;
      21 20 65 68 74 20 67 6E 69 7A 007C5                ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58 007CF     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 007DE                ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20 007EB P.ABP:  .ASCII  \   - DDB_Ptr [EVRLA_0] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 30 5F 41 4C 52 56 007FA                ;
      2F 21 29 00809                ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43 0080C P.ABQ:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74 0081B                ;
      21 20 65 68 74 20 67 6E 69 7A 0082A                ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58 00834     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 00843                ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20 00850 P.ABR:  .ASCII  \   - DDB_Ptr [EVRLA_1] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 31 5F 41 4C 52 56 0085F                ;
      2F 21 29 0086E                ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43 00871 P.ABS:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74 00880                ;
      21 20 65 68 74 20 67 6E 69 7A 0088F                ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58 00899     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 008A8                ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20 008B5 P.ABT:  .ASCII  \   - DDB_Ptr [EVDLB  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 42 4C 44 56 008C4                ;
      2F 21 29 008D3                ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43 008D6 P.ABU:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74 008E5                ;
      21 20 65 68 74 20 67 6E 69 7A 008F4                ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58 008FE     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 0090D                ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20 0091A P.ABV:  .ASCII  \   - DDB_Ptr [EVDLC  ] = !XL(X)!/\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 43 4C 44 56 00929                ;
      2F 21 29 00938                ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43 0093B P.ABW:  .ASCII  \C*** Init_DDB_Status: Initializing the !\  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74 0094A                ;
      21 20 65 68 74 20 67 6E 69 7A 00959                ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58 00963     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 00972                ;
45 5B 20 72 74 50 5F 42 44 44 20 2D 20 20 20 0097F P.ABX:  .ASCII  \   - DDB_Ptr [EVDLD  ] = !XL(X)./\       ;
58 28 4C 58 21 20 3D 20 5D 20 20 44 4C 44 56 0098E                ;
      2F 21 29 0099D                ;
53 5F 42 44 44 5F 74 69 6E 49 20 2A 2A 2A 43 009A0 P.ABY:  .ASCII  \C*** Init_DDB_Status: Initia 'zing the  \  ;
69 6C 61 69 74 69 6E 49 20 3A 73 75 74 61 74 009AF                ;
      21 20 65 68 74 20 67 6E 69 7A 009BE                ;
62 20 73 75 74 61 74 73 20 42 44 44 20 4C 58 009C8     .ASCII  \XL DDB status bits to: !XL!/\       ;
 2F 21 4C 58 21 20 3A 6F 74 20 73 74 69 009D7                ;
```

```
          DSA$AL_APTMAIL=     65024
          DSA$GL_FLAGS=       65024
          DSA$GL_APTCOM=      65028
          DSA$GL_PASSES=      65032
          DSA$GL_UNITS=       65036
          DSA$GL_SECTNO=      65040
          DSA$GL_SID=       65044
          DSA$GL_MSGTYP=      65088
          DSA$GL_ERRNO=       65092
          DSA$GL_EVENT=       65096
          DSA$GL_SUBTNO=      65100
          DSA$GL_TESTNO=      65104
          DSA$GL_PASSNO=      65108
          DSA$GL_DEVLEN=      65112
          DSA$GL_DEVNAM=      65116
          DSA$GL_MSGPTR=      65128
          $MODULE=        P.AAA
            .EXTRN  CRD$BUILD_DDBS, CRD$DISPOSE
            .EXTRN  CRD$INIT_DDB_STATUS
            .EXTRN  CRD$NEW, CRD$GB_CRD_DEBUG
            .EXTRN  CRD$INTERNAL_ERROR
            .EXTRN  CRD$GB_BASE_SYSTEM
            .EXTRN  CRD$PRINT, CRD$AL_IDC_HDWR_SPRT_TABLE
            .EXTRN  CRD$AL_LESI_HDWR_SPRT_TABLE
            .EXTRN  CRD$AL_UDA_0_HDWR_SPRT_TABLE
            .EXTRN  CRD$AL_UDA_1_HDWR_SPRT_TABLE
            .EXTRN  CRD$AL_UDA_2_HDWR_SPRT_TABLE
            .EXTRN  CRD$AL_UDA_3_HDWR_SPRT_TABLE

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              000C 00000     .ENTRY  CRD$BUILD_DDBS, Save R2,R3        ; 0149
           5E       08  C2 00002     SUBL2   #8, SP                  ;
           52 00000000G  EF  9A 00005     MOVZBL  CRD$GB_BASE_SYSTEM, R2        ; 0221
           01       52  91 0000C     CMPB    R2, #1                  ; 0223
           03  13 0000F     BEQL    1$                              ;
         027F  31 00011     BRW     17$                             ;
      04   AE       2C  D0 00014 1$:   MOVL    #44, SIZE             ; 0230
           5E  DD 00018     PUSHL   SP                              ;
      08   AE  9F 0001A     PUSHAB  SIZE                            ;
       00000000V  EF       02  FB 0001D     CALLS   #2, CRD$NEW             ;
           0C       50  E8 00024     BLBS    R0, 2$                  ;
           7E  7C 00027     CLRQ    -(SP)                           ;
           7E       01  CE 00029     MNEGL   #1, -(SP)               ;
       00000000G  EF       03  FB 0002C     CALLS   #3, CRD$INTERNAL_ERROR          ;
           13 00000000G  EF  E9 00033 2$:   BLBC    CRD$GB_CRD_DEBUG, 3$           ;
           6E  DD 0003A     PUSHL   DDB_PTR                         ;
       00000000'  EF  9F 0003C     PUSHAB  P.AAB                   ;
           01  DD 00042     PUSHL   #1                              ;
           1A  DD 00044     PUSHL   #26                             ;
       00000000G  FF       04  FB 00046     CALLS   #4, @CRD$PRINT          ;
           52       6E  D0 0004D 3$:   MOVL    DDB_PTR, R2             ; 0232
           62       52  D0 00050     MOVL    R2, (R2)                ;
      04   A2       52  D0 00053     MOVL    R2, 4(R2)               ; 0233
```

```
      0C    A2 00000000G  EF  9E 00057    MOVAB   CRD$AL_IDC_HDWR_SPRT_TABLE, 12(R2)       ; 0235
      10    A2  D4 0005F    CLRL    16(R2)          ; 0236
      08    A2  D4 00062    CLRL    8(R2)           ; 0237
            52  DD 00065    PUSHL   R2          ; 0239
       00000000V  EF       01  FB 00067    CALLS   #1, CRD$INIT_DDB_STATUS       ;
      14    A2       0C  88 0006E    BISB2   #12, 20(R2)         ; 0244
      17    A2 04    AE  90 00072    MOVB    SIZE, 23(R2)          ; 0249
            16 00000000G  EF  E9 00077    BLBC    CRD$GB_CRD_DEBUG, 4$          ; 0252
      14    A0  DD 0007E    PUSHL   20(DDB_PTR)         ;
            50  DD 00081    PUSHL   DDB_PTR         ;
       00000000'  EF  9F 00083    PUSHAB  P.AAC           ;
            01  DD 00089    PUSHL   #1          ;
            1A  DD 0008B    PUSHL   #26         ;
       00000000G  FF       05  FB 0008D    CALLS   #5, @CRD$PRINT        ;
            53       52  D0 00094 4$:   MOVL    R2, PREV_DDB_PTR          ; 0253
       00000000'  EF       52  D0 00097    MOVL    R2, CRD$GA_FIRST_DIAGNOSTIC       ; 0255
       00000000'  EF       52  D0 0009E    MOVL    R2, CRD$GA_CURRENT_DIAGNOSTIC       ; 0258
      04    AE       2C  D0 000A5    MOVL    #44, SIZE          ; 0265
            5E  DD 000A9    PUSHL   SP          ;
      08    AE  9F 000AB    PUSHAB  SIZE            ;
       00000000V  EF       02  FB 000AE    CALLS   #2, CRD$NEW           ;
            0C       50  E8 000B5    BLBS    R0, 5$          ;
            7E  7C 000B8    CLRQ    -(SP)           ;
            7E       01  CE 000BA    MNEGL   #1, -(SP)           ;
       00000000G  EF       03  FB 000BD    CALLS   #3, CRD$INTERNAL_ERROR        ;
            13 00000000G  EF  E9 000C4 5$:   BLBC    CRD$GB_CRD_DEBUG, 6$          ;
            6E  DD 000CB    PUSHL   DDB_PTR         ;
       00000000'  EF  9F 000CD    PUSHAB  P.AAD           ;
            01  DD 000D3    PUSHL   #1          ;
            1A  DD 000D5    PUSHL   #26         ;
       00000000G  FF       04  FB 000D7    CALLS   #4, @CRD$PRINT        ;
            63 00    BE  0E 000DE 6$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)          ; 0267
            52       6E  D0 000E2    MOVL    DDB_PTR, R2          ; 0269
      0C    A2 00000000G  EF  9E 000E5    MOVAB   CRD$AL_IDC_HDWR_SPRT_TABLE+32, 12(R2)       ;
      10    A2  D4 000ED    CLRL    16(R2)          ; 0270
      08    A2  D4 000F0    CLRL    8(R2)           ; 0271
            52  DD 000F3    PUSHL   R2          ; 0273
       00000000V  EF       01  FB 000F5    CALLS   #1, CRD$INIT_DDB_STATUS       ;
      14    A2       04  88 000FC    BISB2   #4, 20(R2)          ; 0275
      17    A2 04    AE  90 00100    MOVB    SIZE, 23(R2)          ; 0278
            16 00000000G  EF  E9 00105    BLBC    CRD$GB_CRD_DEBUG, 7$          ; 0280
      14    A0  DD 0010C    PUSHL   20(DDB_PTR)         ;
            50  DD 0010F    PUSHL   DDB_PTR         ;
       00000000'  EF  9F 00111    PUSHAB  P.AAE           ;
            01  DD 00117    PUSHL   #1          ;
            1A  DD 00119    PUSHL   #26         ;
       00000000G  FF       05  FB 0011B    CALLS   #5, @CRD$PRINT        ;
            53       52  D0 00122 7$:   MOVL    R2, PREV_DDB_PTR          ; 0281
      04    AE       2C  D0 00125    MOVL    #44, SIZE          ; 0287
            5E  DD 00129    PUSHL   SP          ;
      08    AE  9F 0012B    PUSHAB  SIZE            ;
       00000000V  EF       02  FB 0012E    CALLS   #2, CRD$NEW           ;
            0C       50  E8 00135    BLBS    R0, 8$          ;
            7E  7C 00138    CLRQ    -(SP)           ;
            7E       01  CE 0013A    MNEGL   #1, -(SP)           ;
```

```
                00000000G  EF       03  FB 0013D    CALLS   #3, CRD$INTERNAL_ERROR        ;
                    13 00000000G  EF  E9 00144 8$:   BLBC    CRD$GB_CRD_DEBUG, 9$          ;
                    6E  DD 0014B    PUSHL   DDB_PTR          ;
                00000000'  EF  9F 0014D    PUSHAB  P.AAF           ;
                    01  DD 00153    PUSHL   #1            ;
                    1A  DD 00155    PUSHL   #26           ;
                00000000G  FF       04  FB 00157    CALLS   #4, @CRD$PRINT          ;
                    63 00   BE  0E 0015E 9$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0289
                    52      6E  D0 00162    MOVL    DDB_PTR, R2           ; 0291
                0C  A2 00000000G  EF  9E 00165    MOVAB   CRD$AL_IDC_HDWR_SPRT_TABLE+64, 12(R2)       ;
                10  A2  D4 0016D    CLRL    16(R2)          ; 0292
                08  A2  D4 00170    CLRL    8(R2)           ; 0293
                    52  DD 00173    PUSHL   R2            ; 0295
                00000000V  EF       01  FB 00175    CALLS   #1, CRD$INIT_DDB_STATUS        ;
                17  A2 04   AE  90 0017C    MOVB    SIZE, 23(R2)          ; 0297
                    16 00000000G  EF  E9 00181    BLBC    CRD$GB_CRD_DEBUG, 10$         ; 0300
                14  A0  DD 00188    PUSHL   20(DDB_PTR)          ;
                    50  DD 0018B    PUSHL   DDB_PTR          ;
                00000000'  EF  9F 0018D    PUSHAB  P.AAG           ;
                    01  DD 00193    PUSHL   #1            ;
                    1A  DD 00195    PUSHL   #26           ;
                00000000G  FF       05  FB 00197    CALLS   #5, @CRD$PRINT          ;
                    53      52  D0 0019E 10$:   MOVL    R2, PREV_DDB_PTR          ; 0301
                04  AE      2C  D0 001A1    MOVL    #44, SIZE           ; 0307
                    5E  DD 001A5    PUSHL   SP            ;
                08  AE  9F 001A7    PUSHAB  SIZE          ;
                00000000V  EF       02  FB 001AA    CALLS   #2, CRD$NEW           ;
                    0C      50  E8 001B1    BLBS    R0, 11$           ;
                    7E  7C 001B4    CLRQ    -(SP)           ;
                    7E      01  CE 001B6    MNEGL   #1, -(SP)         ;
                00000000G  EF       03  FB 001B9    CALLS   #3, CRD$INTERNAL_ERROR        ;
                    13 00000000G  EF  E9 001C0 11$:   BLBC    CRD$GB_CRD_DEBUG, 12$         ;
                    6E  DD 001C7    PUSHL   DDB_PTR          ;
                00000000'  EF  9F 001C9    PUSHAB  P.AAH           ;
                    01  DD 001CF    PUSHL   #1            ;
                    1A  DD 001D1    PUSHL   #26           ;
                00000000G  FF       04  FB 001D3    CALLS   #4, @CRD$PRINT          ;
                    63 00   BE  0E 001DA 12$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0309
                    52      6E  D0 001DE    MOVL    DDB_PTR, R2           ; 0311
                0C  A2 00000000G  EF  9E 001E1    MOVAB   CRD$AL_IDC_HDWR_SPRT_TABLE+96, 12(R2)       ;
                10  A2  D4 001E9    CLRL    16(R2)          ; 0312
                08  A2  D4 001EC    CLRL    8(R2)           ; 0313
                    52  DD 001EF    PUSHL   R2            ; 0315
                00000000V  EF       01  FB 001F1    CALLS   #1, CRD$INIT_DDB_STATUS        ;
                14  A2      04  88 001F8    BISB2   #4, 20(R2)          ; 0317
                17  A2 04   AE  90 001FC    MOVB    SIZE, 23(R2)          ; 0320
                    16 00000000G  EF  E9 00201    BLBC    CRD$GB_CRD_DEBUG, 13$         ; 0323
                14  A0  DD 00208    PUSHL   20(DDB_PTR)          ;
                    50  DD 0020B    PUSHL   DDB_PTR          ;
                00000000'  EF  9F 0020D    PUSHAB  P.AAI           ;
                    01  DD 00213    PUSHL   #1            ;
                    1A  DD 00215    PUSHL   #26           ;
                00000000G  FF       05  FB 00217    CALLS   #5, @CRD$PRINT          ;
                    53      52  D0 0021E 13$:   MOVL    R2, PREV_DDB_PTR          ; 0324
                04  AE      2C  D0 00221    MOVL    #44, SIZE           ; 0330
```

```
                5E  DD 00225     PUSHL   SP                      ;
            08  AE  9F 00227     PUSHAB  SIZE                    ;
             00000000V  EF        02  FB 0022A     CALLS   #2, CRD$NEW                 ;
                0C       50  E8 00231     BLBS    R0, 14$                 ;
                7E  7C 00234     CLRQ    -(SP)                   ;
                7E       01  CE 00236     MNEGL   #1, -(SP)               ;
             00000000G  EF        03  FB 00239     CALLS   #3, CRD$INTERNAL_ERROR          ;
                13 00000000G  EF  E9 00240 14$:   BLBC    CRD$GB_CRD_DEBUG, 15$           ;
                6E  DD 00247     PUSHL   DDB_PTR                 ;
             00000000'  EF  9F 00249     PUSHAB  P.AAJ                   ;
                01  DD 0024F     PUSHL   #1                      ;
                1A  DD 00251     PUSHL   #26                     ;
             00000000G  FF        04  FB 00253     CALLS   #4, @CRD$PRINT          ;
                63 00    BE  0E 0025A 15$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0332
                52       6E  D0 0025E     MOVL    DDB_PTR, R2             ; 0334
            0C  A2 00000000G  EF  9E 00261     MOVAB   CRD$AL_IDC_HDWR_SPRT_TABLE+128, 12(R2)      ;
            10  A2  D4 00269     CLRL    16(R2)                  ; 0335
            08  A2  D4 0026C     CLRL    8(R2)                   ; 0336
                52  DD 0026F     PUSHL   R2                      ; 0338
             00000000V  EF        01  FB 00271     CALLS   #1, CRD$INIT_DDB_STATUS         ;
            17  A2 04   AE  90 00278     MOVB    SIZE, 23(R2)            ; 0340
                01 00000000G  EF  E8 0027D     BLBS    CRD$GB_CRD_DEBUG, 16$           ; 0343
              04 00284     RET                     ;
            14  A0  DD 00285 16$:   PUSHL   20(DDB_PTR)             ;
                50  DD 00288     PUSHL   DDB_PTR                 ;
             00000000'  EF  9F 0028A     PUSHAB  P.AAK                   ;
               0A20  31 00290     BRW     86$                     ;
                02       52  91 00293 17$:   CMPB    R2, #2              ; 0352
                03  13 00296     BEQL    18$                     ;
               0112  31 00298     BRW     25$                     ;
            04  AE       2C  D0 0029B 18$:   MOVL    #44, SIZE               ; 0358
                5E  DD 0029F     PUSHL   SP                      ;
            08  AE  9F 002A1     PUSHAB  SIZE                    ;
             00000000V  EF        02  FB 002A4     CALLS   #2, CRD$NEW                 ;
                0C       50  E8 002AB     BLBS    R0, 19$                 ;
                7E  7C 002AE     CLRQ    -(SP)                   ;
                7E       01  CE 002B0     MNEGL   #1, -(SP)               ;
             00000000G  EF        03  FB 002B3     CALLS   #3, CRD$INTERNAL_ERROR          ;
                13 00000000G  EF  E9 002BA 19$:   BLBC    CRD$GB_CRD_DEBUG, 20$           ;
                6E  DD 002C1     PUSHL   DDB_PTR                 ;
             00000000'  EF  9F 002C3     PUSHAB  P.AAL                   ;
                01  DD 002C9     PUSHL   #1                      ;
                1A  DD 002CB     PUSHL   #26                     ;
             00000000G  FF        04  FB 002CD     CALLS   #4, @CRD$PRINT          ;
                52       6E  D0 002D4 20$:   MOVL    DDB_PTR, R2             ; 0360
                62       52  D0 002D7     MOVL    R2, (R2)                ;
            04  A2       52  D0 002DA     MOVL    R2, 4(R2)               ; 0361
            0C  A2 00000000G  EF  9E 002DE     MOVAB   CRD$AL_LESI_HDWR_SPRT_TABLE, 12(R2)     ; 0363
            10  A2  D4 002E6     CLRL    16(R2)                  ; 0364
            08  A2  D4 002E9     CLRL    8(R2)                   ; 0365
                52  DD 002EC     PUSHL   R2                      ; 0367
             00000000V  EF        01  FB 002EE     CALLS   #1, CRD$INIT_DDB_STATUS         ;
            14  A2       04  88 002F5     BISB2   #4, 20(R2)              ; 0369
            17  A2 04   AE  90 002F9     MOVB    SIZE, 23(R2)            ; 0373
                16 00000000G  EF  E9 002FE     BLBC    CRD$GB_CRD_DEBUG, 21$           ; 0376
```

```
      14    A0  DD 00305     PUSHL   20(DDB_PTR)          ;
            50  DD 00308     PUSHL   DDB_PTR             ;
             00000000'  EF  9F 0030A     PUSHAB  P.AAM             ;
                 01  DD 00310     PUSHL   #1              ;
                 1A  DD 00312     PUSHL   #26             ;
             00000000G  FF       05  FB 00314     CALLS   #5, @CRD$PRINT          ;
                 53       52  D0 0031B 21$:    MOVL    R2, PREV_DDB_PTR        ; 0377
             00000000'  EF       52  D0 0031E     MOVL    R2, CRD$GA_FIRST_DIAGNOSTIC         ; 0379
             00000000'  EF       52  D0 00325     MOVL    R2, CRD$GA_CURRENT_DIAGNOSTIC       ; 0382
      04    AE        2C  D0 0032C     MOVL    #44, SIZE           ; 0389
            5E  DD 00330     PUSHL   SP              ;
      08    AE  9F 00332     PUSHAB  SIZE            ;
             00000000V  EF       02  FB 00335     CALLS   #2, CRD$NEW             ;
                 0C       50  E8 0033C     BLBS    R0, 22$             ;
                 7E  7C 0033F     CLRQ    -(SP)           ;
                 7E       01  CE 00341     MNEGL   #1, -(SP)           ;
             00000000G  EF       03  FB 00344     CALLS   #3, CRD$INTERNAL_ERROR          ;
                 13 00000000G  EF  E9 0034B 22$:    BLBC    CRD$GB_CRD_DEBUG, 23$           ;
                 6E  DD 00352     PUSHL   DDB_PTR             ;
             00000000'  EF  9F 00354     PUSHAB  P.AAN             ;
                 01  DD 0035A     PUSHL   #1              ;
                 1A  DD 0035C     PUSHL   #26             ;
             00000000G  FF       04  FB 0035E     CALLS   #4, @CRD$PRINT          ;
                 63 00    BE  0E 00365 23$:    INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0391
                 52       6E  D0 00369     MOVL    DDB_PTR, R2         ; 0393
      0C    A2 00000000G  EF  9E 0036C     MOVAB   CRD$AL_LESI_HDWR_SPRT_TABLE+32, 12(R2)      ;
      10    A2  D4 00374     CLRL    16(R2)              ; 0394
      08    A2  D4 00377     CLRL    8(R2)               ; 0395
            52  DD 0037A     PUSHL   R2              ; 0397
             00000000V  EF       01  FB 0037C     CALLS   #1, CRD$INIT_DDB_STATUS         ;
      14    A2       04  88 00383     BISB2   #4, 20(R2)          ; 0399
      17    A2 04    AE  90 00387     MOVB    SIZE, 23(R2)            ; 0402
                 16 00000000G  EF  E9 0038C     BLBC    CRD$GB_CRD_DEBUG, 24$           ; 0404
      14    A0  DD 00393     PUSHL   20(DDB_PTR)         ;
            50  DD 00396     PUSHL   DDB_PTR             ;
             00000000'  EF  9F 00398     PUSHAB  P.AAO             ;
                 01  DD 0039E     PUSHL   #1              ;
                 1A  DD 003A0     PUSHL   #26             ;
             00000000G  FF       05  FB 003A2     CALLS   #5, @CRD$PRINT          ;
                 53       52  D0 003A9 24$:    MOVL    R2, PREV_DDB_PTR        ; 0405
              04 003AC     RET                 ; 0221
                 03       52  91 003AD 25$:    CMPB    R2, #3          ; 0446
                 03  13 003B0     BEQL    26$             ;
               01FB  31 003B2     BRW     39$             ;
      04    AE        2C  D0 003B5 26$:    MOVL    #44, SIZE           ; 0452
            5E  DD 003B9     PUSHL   SP              ;
      08    AE  9F 003BB     PUSHAB  SIZE            ;
             00000000V  EF       02  FB 003BE     CALLS   #2, CRD$NEW             ;
                 0C       50  E8 003C5     BLBS    R0, 27$             ;
                 7E  7C 003C8     CLRQ    -(SP)           ;
                 7E       01  CE 003CA     MNEGL   #1, -(SP)           ;
             00000000G  EF       03  FB 003CD     CALLS   #3, CRD$INTERNAL_ERROR          ;
                 13 00000000G  EF  E9 003D4 27$:    BLBC    CRD$GB_CRD_DEBUG, 28$           ;
                 6E  DD 003DB     PUSHL   DDB_PTR             ;
             00000000'  EF  9F 003DD     PUSHAB  P.AAP             ;
```

```
            01  DD 003E3     PUSHL   #1              ;
            1A  DD 003E5     PUSHL   #26              ;
 00000000G  FF       04  FB 003E7     CALLS   #4, @CRD$PRINT          ;
            52       6E  D0 003EE 28$:   MOVL    DDB_PTR, R2          ; 0454
            62       52  D0 003F1     MOVL    R2, (R2)            ;
      04    A2       52  D0 003F4     MOVL    R2, 4(R2)            ; 0455
      0C    A2 00000000G  EF  9E 003F8    MOVAB   CRD$AL_UDA_0_HDWR_SPRT_TABLE, 12(R2)       ; 0457
      10    A2  D4 00400     CLRL    16(R2)              ; 0458
      08    A2  D4 00403     CLRL    8(R2)               ; 0459
            52  DD 00406     PUSHL   R2              ; 0461
 00000000V  EF       01  FB 00408     CALLS   #1, CRD$INIT_DDB_STATUS         ;
      14    A2       0C  88 0040F     BISB2   #12, 20(R2)             ; 0466
      17    A2 04    AE  90 00413     MOVB    SIZE, 23(R2)            ; 0471
            16 00000000G  EF  E9 00418    BLBC    CRD$GB_CRD_DEBUG, 29$           ; 0474
      14    A0  DD 0041F     PUSHL   20(DDB_PTR)             ;
            50  DD 00422     PUSHL   DDB_PTR             ;
 00000000'  EF  9F 00424     PUSHAB  P.AAQ               ;
            01  DD 0042A     PUSHL   #1              ;
            1A  DD 0042C     PUSHL   #26              ;
 00000000G  FF       05  FB 0042E     CALLS   #5, @CRD$PRINT          ;
            53       52  D0 00435 29$:   MOVL    R2, PREV_DDB_PTR            ; 0475
 00000000'  EF       52  D0 00438     MOVL    R2, CRD$GA_FIRST_DIAGNOSTIC         ; 0477
 00000000'  EF       52  D0 0043F     MOVL    R2, CRD$GA_CURRENT_DIAGNOSTIC           ; 0480
      04    AE       2C  D0 00446     MOVL    #44, SIZE           ; 0488
            5E  DD 0044A     PUSHL   SP              ;
      08    AE  9F 0044C     PUSHAB  SIZE            ;
 00000000V  EF       02  FB 0044F     CALLS   #2, CRD$NEW             ;
            0C       50  E8 00456     BLBS    R0, 30$             ;
            7E  7C 00459     CLRQ    -(SP)               ;
            7E       01  CE 0045B     MNEGL   #1, -(SP)           ;
 00000000G  EF       03  FB 0045E     CALLS   #3, CRD$INTERNAL_ERROR          ;
            13 00000000G  EF  E9 00465 30$:   BLBC    CRD$GB_CRD_DEBUG, 31$           ;
            6E  DD 0046C     PUSHL   DDB_PTR             ;
 00000000'  EF  9F 0046E     PUSHAB  P.AAR               ;
            01  DD 00474     PUSHL   #1              ;
            1A  DD 00476     PUSHL   #26              ;
 00000000G  FF       04  FB 00478     CALLS   #4, @CRD$PRINT          ;
            63 00    BE  0E 0047F 31$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)          ; 0490
            52       6E  D0 00483     MOVL    DDB_PTR, R2             ; 0492
      0C    A2 00000000G  EF  9E 00486    MOVAB   CRD$AL_UDA_0_HDWR_SPRT_TABLE+32, 12(R2)     ;
      10    A2  D4 0048E     CLRL    16(R2)              ; 0493
      08    A2  D4 00491     CLRL    8(R2)               ; 0494
            52  DD 00494     PUSHL   R2              ; 0496
 00000000V  EF       01  FB 00496     CALLS   #1, CRD$INIT_DDB_STATUS         ;
      17    A2 04    AE  90 0049D     MOVB    SIZE, 23(R2)            ; 0498
            16 00000000G  EF  E9 004A2    BLBC    CRD$GB_CRD_DEBUG, 32$           ; 0501
      14    A0  DD 004A9     PUSHL   20(DDB_PTR)             ;
            50  DD 004AC     PUSHL   DDB_PTR             ;
 00000000'  EF  9F 004AE     PUSHAB  P.AAS               ;
            01  DD 004B4     PUSHL   #1              ;
            1A  DD 004B6     PUSHL   #26              ;
 00000000G  FF       05  FB 004B8     CALLS   #5, @CRD$PRINT          ;
            53       52  D0 004BF 32$:   MOVL    R2, PREV_DDB_PTR            ; 0502
      04    AE       2C  D0 004C2     MOVL    #44, SIZE           ; 0508
            5E  DD 004C6     PUSHL   SP              ;
```

```
      08   AE  9F 004C8    PUSHAB  SIZE                ;
       00000000V  EF       02  FB 004CB    CALLS   #2, CRD$NEW              ;
           0C      50  E8 004D2    BLBS    R0, 33$              ;
           7E  7C 004D5    CLRQ    -(SP)               ;
           7E      01  CE 004D7    MNEGL   #1, -(SP)           ;
       00000000G  EF       03  FB 004DA    CALLS   #3, CRD$INTERNAL_ERROR          ;
           13 00000000G  EF  E9 004E1 33$:   BLBC    CRD$GB_CRD_DEBUG, 34$           ;
           6E  DD 004E8    PUSHL   DDB_PTR             ;
       00000000'  EF  9F 004EA    PUSHAB  P.AAT               ;
           01  DD 004F0    PUSHL   #1              ;
           1A  DD 004F2    PUSHL   #26             ;
       00000000G  FF       04  FB 004F4    CALLS   #4, @CRD$PRINT          ;
           63 00   BE  0E 004FB 34$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0510
           52      6E  D0 004FF    MOVL    DDB_PTR, R2             ; 0512
      0C   A2 00000000G  EF  9E 00502    MOVAB   CRD$AL_UDA_0_HDWR_SPRT_TABLE+64, 12(R2)     ;
      10   A2  D4 0050A    CLRL    16(R2)              ; 0513
      08   A2  D4 0050D    CLRL    8(R2)               ; 0514
           52  DD 00510    PUSHL   R2              ; 0516
       00000000V  EF       01  FB 00512    CALLS   #1, CRD$INIT_DDB_STATUS         ;
      17   A2 04   AE  90 00519    MOVB    SIZE, 23(R2)            ; 0519
           16 00000000G  EF  E9 0051E    BLBC    CRD$GB_CRD_DEBUG, 35$           ; 0522
      14   A0  DD 00525    PUSHL   20(DDB_PTR)             ;
           50  DD 00528    PUSHL   DDB_PTR             ;
       00000000'  EF  9F 0052A    PUSHAB  P.AAU               ;
           01  DD 00530    PUSHL   #1              ;
           1A  DD 00532    PUSHL   #26             ;
       00000000G  FF       05  FB 00534    CALLS   #5, @CRD$PRINT          ;
           53      52  D0 0053B 35$:   MOVL    R2, PREV_DDB_PTR            ; 0523
      04   AE      2C  D0 0053E    MOVL    #44, SIZE           ; 0529
           5E  DD 00542    PUSHL   SP              ;
      08   AE  9F 00544    PUSHAB  SIZE                ;
       00000000V  EF       02  FB 00547    CALLS   #2, CRD$NEW              ;
           0C      50  E8 0054E    BLBS    R0, 36$              ;
           7E  7C 00551    CLRQ    -(SP)               ;
           7E      01  CE 00553    MNEGL   #1, -(SP)           ;
       00000000G  EF       03  FB 00556    CALLS   #3, CRD$INTERNAL_ERROR          ;
           13 00000000G  EF  E9 0055D 36$:   BLBC    CRD$GB_CRD_DEBUG, 37$           ;
           6E  DD 00564    PUSHL   DDB_PTR             ;
       00000000'  EF  9F 00566    PUSHAB  P.AAV               ;
           01  DD 0056C    PUSHL   #1              ;
           1A  DD 0056E    PUSHL   #26             ;
       00000000G  FF       04  FB 00570    CALLS   #4, @CRD$PRINT          ;
           63 00   BE  0E 00577 37$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0531
           52      6E  D0 0057B    MOVL    DDB_PTR, R2             ; 0533
      0C   A2 00000000G  EF  9E 0057E    MOVAB   CRD$AL_UDA_0_HDWR_SPRT_TABLE+96, 12(R2)     ;
      10   A2  D4 00586    CLRL    16(R2)              ; 0534
      08   A2  D4 00589    CLRL    8(R2)               ; 0535
           52  DD 0058C    PUSHL   R2              ; 0537
       00000000V  EF       01  FB 0058E    CALLS   #1, CRD$INIT_DDB_STATUS         ;
      17   A2 04   AE  90 00595    MOVB    SIZE, 23(R2)            ; 0539
           01 00000000G  EF  E8 0059A    BLBS    CRD$GB_CRD_DEBUG, 38$           ; 0542
        04 005A1    RET                 ;
      14   A0  DD 005A2 38$:   PUSHL   20(DDB_PTR)             ;
           50  DD 005A5    PUSHL   DDB_PTR             ;
       00000000'  EF  9F 005A7    PUSHAB  P.AAW               ;
```

```
          0703  31 005AD     BRW     86$                ;
            04       52  91 005B0 39$:   CMPB    R2, #4          ; 0550
            03  13 005B3     BEQL    40$                ;
          027B  31 005B5     BRW     56$                ;
      04    AE       2C  D0 005B8 40$:   MOVL    #44, SIZE          ; 0556
            5E  DD 005BC     PUSHL   SP                 ;
      08    AE  9F 005BE     PUSHAB  SIZE               ;
       00000000V  EF       02  FB 005C1    CALLS   #2, CRD$NEW           ;
            0C       50  E8 005C8     BLBS    R0, 41$            ;
            7E  7C 005CB     CLRQ    -(SP)              ;
            7E       01  CE 005CD     MNEGL   #1, -(SP)          ;
       00000000G  EF       03  FB 005D0    CALLS   #3, CRD$INTERNAL_ERROR        ;
            13 00000000G  EF  E9 005D7 41$:   BLBC    CRD$GB_CRD_DEBUG, 42$         ;
            6E  DD 005DE     PUSHL   DDB_PTR            ;
       00000000'  EF  9F 005E0    PUSHAB  P.AAX              ;
            01  DD 005E6     PUSHL   #1                 ;
            1A  DD 005E8     PUSHL   #26                ;
       00000000G  FF       04  FB 005EA    CALLS   #4, @CRD$PRINT         ;
            52       6E  D0 005F1 42$:   MOVL    DDB_PTR, R2           ; 0558
            62       52  D0 005F4     MOVL    R2, (R2)           ;
      04    A2       52  D0 005F7     MOVL    R2, 4(R2)          ; 0559
      0C    A2 00000000G  EF  9E 005FB    MOVAB   CRD$AL_UDA_1_HDWR_SPRT_TABLE, 12(R2)       ; 0561
      10    A2  D4 00603     CLRL    16(R2)             ; 0562
      08    A2  D4 00606     CLRL    8(R2)              ; 0563
            52  DD 00609     PUSHL   R2                 ; 0565
       00000000V  EF       01  FB 0060B    CALLS   #1, CRD$INIT_DDB_STATUS        ;
      14    A2       0C  88 00612     BISB2   #12, 20(R2)           ; 0570
      17    A2 04    AE  90 00616     MOVB    SIZE, 23(R2)          ; 0575
            16 00000000G  EF  E9 0061B    BLBC    CRD$GB_CRD_DEBUG, 43$         ; 0578
      14    A0  DD 00622     PUSHL   20(DDB_PTR)        ;
            50  DD 00625     PUSHL   DDB_PTR            ;
       00000000'  EF  9F 00627    PUSHAB  P.AAY              ;
            01  DD 0062D     PUSHL   #1                 ;
            1A  DD 0062F     PUSHL   #26                ;
       00000000G  FF       05  FB 00631    CALLS   #5, @CRD$PRINT         ;
            53       52  D0 00638 43$:   MOVL    R2, PREV_DDB_PTR          ; 0579
       00000000'  EF       52  D0 0063B    MOVL    R2, CRD$GA_FIRST_DIAGNOSTIC        ; 0581
       00000000'  EF       52  D0 00642    MOVL    R2, CRD$GA_CURRENT_DIAGNOSTIC        ; 0584
      04    AE       2C  D0 00649     MOVL    #44, SIZE          ; 0591
            5E  DD 0064D     PUSHL   SP                 ;
      08    AE  9F 0064F     PUSHAB  SIZE               ;
       00000000V  EF       02  FB 00652    CALLS   #2, CRD$NEW           ;
            0C       50  E8 00659     BLBS    R0, 44$            ;
            7E  7C 0065C     CLRQ    -(SP)              ;
            7E       01  CE 0065E     MNEGL   #1, -(SP)          ;
       00000000G  EF       03  FB 00661    CALLS   #3, CRD$INTERNAL_ERROR        ;
            13 00000000G  EF  E9 00668 44$:   BLBC    CRD$GB CRD_DEBUG, 45$         ;
            6E  DD 0066F     PUSHL   DDB_PTR            ;
       00000000'  EF  9F 00671    PUSHAB  P.AAZ              ;
            01  DD 00677     PUSHL   #1                 ;
            1A  DD 00679     PUSHL   #26                ;
       00000000G  FF       04  FB 0067B    CALLS   #4, @CRD$PRINT         ;
            63 00    BE  0E 00682 45$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0593
            52       6E  D0 00686     MOVL    DDB_PTR, R2           ; 0595
      0C    A2 00000000G  EF  9E 00689    MOVAB   CRD$AL_UDA_1_HDWR_SPRT_TABLE+32, 12(R2)     ;
```

```
      10   A2  D4 00691     CLRL    16(R2)           ; 0596
      08   A2  D4 00694     CLRL    8(R2)            ; 0597
           52  DD 00697     PUSHL   R2             ; 0599
       00000000V  EF       01  FB 00699     CALLS   #1, CRD$INIT_DDB_STATUS        ;
      14   A2       04  88 006A0     BISB2   #4, 20(R2)          ; 0601
      17   A2 04    AE  90 006A4     MOVB    SIZE, 23(R2)          ; 0604
           16 00000000G  EF  E9 006A9     BLBC    CRD$GB_CRD_DEBUG, 46$          ; 0606
      14   A0  DD 006B0     PUSHL   20(DDB_PTR)          ;
           50  DD 006B3     PUSHL   DDB_PTR          ;
       00000000'  EF  9F 006B5     PUSHAB  P.ABA          ;
           01  DD 006BB     PUSHL   #1             ;
           1A  DD 006BD     PUSHL   #26            ;
       00000000G  FF       05  FB 006BF     CALLS   #5, @CRD$PRINT          ;
           53       52  D0 006C6 46$:   MOVL    R2, PREV_DDB_PTR          ; 0607
      04   AE       2C  D0 006C9     MOVL    #44, SIZE           ; 0613
           5E  DD 006CD     PUSHL   SP             ;
      08   AE  9F 006CF     PUSHAB  SIZE           ;
       00000000V  EF       02  FB 006D2     CALLS   #2, CRD$NEW          ;
           0C       50  E8 006D9     BLBS    R0, 47$          ;
           7E  7C 006DC     CLRQ    -(SP)          ;
           7E       01  CE 006DE     MNEGL   #1, -(SP)          ;
       00000000G  EF       03  FB 006E1     CALLS   #3, CRD$INTERNAL_ERROR        ;
           13 00000000G  EF  E9 006E8 47$:   BLBC    CRD$GB_CRD_DEBUG, 48$          ;
           6E  DD 006EF     PUSHL   DDB_PTR          ;
       00000000'  EF  9F 006F1     PUSHAB  P.ABB          ;
           01  DD 006F7     PUSHL   #1             ;
           1A  DD 006F9     PUSHL   #26            ;
       00000000G  FF       04  FB 006FB     CALLS   #4, @CRD$PRINT          ;
           63 00    BE  0E 00702 48$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0615
           52       6E  D0 00706     MOVL    DDB_PTR, R2           ; 0617
      0C   A2 00000000G  EF  9E 00709     MOVAB   CRD$AL_UDA_1_HDWR_SPRT_TABLE+64, 12(R2)     ;
      10   A2  D4 00711     CLRL    16(R2)           ; 0618
      08   A2  D4 00714     CLRL    8(R2)            ; 0619
           52  DD 00717     PUSHL   R2             ; 0621
       00000000V  EF       01  FB 00719     CALLS   #1, CRD$INIT_DDB_STATUS        ;
      17   A2 04    AE  90 00720     MOVB    SIZE, 23(R2)          ; 0623
           16 00000000G  EF  E9 00725     BLBC    CRD$GB_CRD_DEBUG, 49$          ; 0626
      14   A0  DD 0072C     PUSHL   20(DDB_PTR)          ;
           50  DD 0072F     PUSHL   DDB_PTR          ;
       00000000'  EF  9F 00731     PUSHAB  P.ABC          ;
           01  DD 00737     PUSHL   #1             ;
           1A  DD 00739     PUSHL   #26            ;
       00000000G  FF       05  FB 0073B     CALLS   #5, @CRD$PRINT          ;
           53       52  D0 00742 49$:   MOVL    R2, PREV_DDB_PTR          ; 0627
      04   AE       2C  D0 00745     MOVL    #44, SIZE           ; 0633
           5E  DD 00749     PUSHL   SP             ;
      08   AE  9F 0074B     PUSHAB  SIZE           ;
       00000000V  EF       02  FB 0074E     CALLS   #2, CRD$NEW          ;
           0C       50  E8 00755     BLBS    R0, 50$          ;
           7E  7C 00758     CLRQ    -(SP)          ;
           7E       01  CE 0075A     MNEGL   #1, -(SP)          ;
       00000000G  EF       03  FB 0075D     CALLS   #3, CRD$INTERNAL_ERROR        ;
           13 00000000G  EF  E9 00764 50$:   BLBC    CRD$GB_CRD_DEBUG, 51$          ;
           6E  DD 0076B     PUSHL   DDB_PTR          ;
       00000000'  EF  9F 0076D     PUSHAB  P.ABD          ;
```

```
            01  DD 00773     PUSHL   #1              ;
            1A  DD 00775     PUSHL   #26             ;
 00000000G  FF       04  FB 00777     CALLS   #4, @CRD$PRINT          ;
            63 00    BE  0E 0077E 51$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0635
            52       6E  D0 00782     MOVL    DDB_PTR, R2            ; 0637
        0C  A2 00000000G  EF  9E 00785     MOVAB   CRD$AL_UDA_1_HDWR_SPRT_TABLE+96, 12(R2)     ;
        10  A2  D4 0078D     CLRL    16(R2)              ; 0638
        08  A2  D4 00790     CLRL    8(R2)               ; 0639
            52  DD 00793     PUSHL   R2              ; 0641
 00000000V  EF       01  FB 00795     CALLS   #1, CRD$INIT_DDB_STATUS         ;
        17  A2 04   AE  90 0079C     MOVB    SIZE, 23(R2)            ; 0644
            16 00000000G  EF  E9 007A1     BLBC    CRD$GB_CRD_DEBUG, 52$           ; 0647
        14  A0  DD 007A8     PUSHL   20(DDB_PTR)             ;
            50  DD 007AB     PUSHL   DDB_PTR             ;
 00000000'  EF  9F 007AD     PUSHAB  P.ABE               ;
            01  DD 007B3     PUSHL   #1              ;
            1A  DD 007B5     PUSHL   #26             ;
 00000000G  FF       05  FB 007B7     CALLS   #5, @CRD$PRINT          ;
            53       52  D0 007BE 52$:   MOVL    R2, PREV_DDB_PTR           ; 0648
        04  AE       2C  D0 007C1     MOVL    #44, SIZE              ; 0654
            5E  DD 007C5     PUSHL   SP              ;
        08  AE  9F 007C7     PUSHAB  SIZE            ;
 00000000V  EF       02  FB 007CA     CALLS   #2, CRD$NEW             ;
            0C       50  E8 007D1     BLBS    R0, 53$             ;
            7E  7C 007D4     CLRQ    -(SP)               ;
            7E       01  CE 007D6     MNEGL   #1, -(SP)               ;
 00000000G  EF       03  FB 007D9     CALLS   #3, CRD$INTERNAL_ERROR          ;
            13 00000000G  EF  E9 007E0 53$:   BLBC    CRD$GB_CRD_DEBUG, 54$           ;
            6E  DD 007E7     PUSHL   DDB_PTR             ;
 00000000'  EF  9F 007E9     PUSHAB  P.ABF               ;
            01  DD 007EF     PUSHL   #1              ;
            1A  DD 007F1     PUSHL   #26             ;
 00000000G  FF       04  FB 007F3     CALLS   #4, @CRD$PRINT          ;
            63 00    BE  0E 007FA 54$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0656
            52       6E  D0 007FE     MOVL    DDB_PTR, R2            ; 0658
        0C  A2 00000000G  EF  9E 00801     MOVAB   CRD$AL_UDA_1_HDWR_SPRT_TABLE+128, 12(R2)    ;
        10  A2  D4 00809     CLRL    16(R2)              ; 0659
        08  A2  D4 0080C     CLRL    8(R2)               ; 0660
            52  DD 0080F     PUSHL   R2              ; 0662
 00000000V  EF       01  FB 00811     CALLS   #1, CRD$INIT_DDB_STATUS         ;
        17  A2 04   AE  90 00818     MOVB    SIZE, 23(R2)            ; 0664
            01 00000000G  EF  E8 0081D     BLBS    CRD$GB_CRD_DEBUG, 55$           ; 0667
          04 00824     RET                 ;
        14  A0  DD 00825 55$:   PUSHL   20(DDB_PTR)             ;
            50  DD 00828     PUSHL   DDB_PTR             ;
 00000000'  EF  9F 0082A     PUSHAB  P.ABG               ;
           0480  31 00830     BRW     86$             ;
            05       52  91 00833 56$:   CMPB    R2, #5              ; 0679
            03  13 00836     BEQL    57$             ;
           01FB  31 00838     BRW     70$             ;
        04  AE       2C  D0 0083B 57$:   MOVL    #44, SIZE              ; 0685
            5E  DD 0083F     PUSHL   SP              ;
        08  AE  9F 00841     PUSHAB  SIZE            ;
 00000000V  EF       02  FB 00844     CALLS   #2, CRD$NEW             ;
            0C       50  E8 0084B     BLBS    R0, 58$             ;
```

```
            7E  7C 0084E     CLRQ    -(SP)                ;
            7E       01  CE 00850    MNEGL   #1, -(SP)          ;
         00000000G   EF       03  FB 00853    CALLS   #3, CRD$INTERNAL_ERROR         ;
            13 00000000G   EF  E9 0085A 58$:   BLBC    CRD$GB_CRD_DEBUG, 59$          ;
            6E  DD 00861     PUSHL   DDB_PTR              ;
         00000000'   EF  9F 00863    PUSHAB  P.ABH              ;
            01  DD 00869     PUSHL   #1               ;
            1A  DD 0086B     PUSHL   #26              ;
         00000000G   FF       04  FB 0086D    CALLS   #4, @CRD$PRINT          ;
            52       6E  D0 00874 59$:   MOVL    DDB_PTR, R2            ; 0687
            62       52  D0 00877    MOVL    R2, (R2)             ;
         04 A2       52  D0 0087A    MOVL    R2, 4(R2)             ; 0688
         0C A2 00000000G   EF  9E 0087E    MOVAB   CRD$AL_UDA_2_HDWR_SPRT_TABLE, 12(R2)        ; 0690
         10 A2  D4 00886     CLRL    16(R2)               ; 0691
         08 A2  D4 00889     CLRL    8(R2)                ; 0692
            52  DD 0088C     PUSHL   R2               ; 0694
         00000000V   EF       01  FB 0088E    CALLS   #1, CRD$INIT_DDB_STATUS         ;
         14 A2       0C  88 00895    BISB2   #12, 20(R2)              ; 0699
         17 A2 04    AE  90 00899    MOVB    SIZE, 23(R2)              ; 0704
            16 00000000G   EF  E9 0089E    BLBC    CRD$GB_CRD_DEBUG, 60$          ; 0707
         14 A0  DD 008A5     PUSHL   20(DDB_PTR)              ;
            50  DD 008A8     PUSHL   DDB_PTR              ;
         00000000'   EF  9F 008AA    PUSHAB  P.ABI              ;
            01  DD 008B0     PUSHL   #1               ;
            1A  DD 008B2     PUSHL   #26              ;
         00000000G   FF       05  FB 008B4    CALLS   #5, @CRD$PRINT          ;
            53       52  D0 008BB 60$:   MOVL    R2, PREV_DDB_PTR           ; 0708
         00000000'   EF       52  D0 008BE    MOVL    R2, CRD$GA_FIRST_DIAGNOSTIC         ; 0710
         00000000'   EF       52  D0 008C5    MOVL    R2, CRD$GA_CURRENT_DIAGNOSTIC         ; 0713
         04 AE       2C  D0 008CC    MOVL    #44, SIZE            ; 0721
            5E  DD 008D0     PUSHL   SP               ;
         08 AE  9F 008D2     PUSHAB  SIZE               ;
         00000000V   EF       02  FB 008D5    CALLS   #2, CRD$NEW            ;
            0C       50  E8 008DC    BLBS    R0, 61$              ;
            7E  7C 008DF     CLRQ    -(SP)                ;
            7E       01  CE 008E1    MNEGL   #1, -(SP)          ;
         00000000G   EF       03  FB 008E4    CALLS   #3, CRD$INTERNAL_ERROR         ;
            13 00000000G   EF  E9 008EB 61$:   BLBC    CRD$GB_CRD_DEBUG, 62$          ;
            6E  DD 008F2     PUSHL   DDB_PTR              ;
         00000000'   EF  9F 008F4    PUSHAB  P.ABJ              ;
            01  DD 008FA     PUSHL   #1               ;
            1A  DD 008FC     PUSHL   #26              ;
         00000000G   FF       04  FB 008FE    CALLS   #4, @CRD$PRINT          ;
            63 00    BE  0E 00905 62$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)         ; 0723
            52       6E  D0 00909    MOVL    DDB_PTR, R2            ; 0725
         0C A2 00000000G   EF  9E 0090C    MOVAB   CRD$AL_UDA_2_HDWR_SPRT_TABLE+32, 12(R2)      ;
         10 A2  D4 00914     CLRL    16(R2)               ; 0726
         08 A2  D4 00917     CLRL    8(R2)                ; 0727
            52  DD 0091A     PUSHL   R2               ; 0729
         00000000V   EF       01  FB 0091C    CALLS   #1, CRD$INIT_DDB_STATUS         ;
         17 A2 04    AE  90 00923    MOVB    SIZE, 23(R2)              ; 0731
            16 00000000G   EF  E9 00928    BLBC    CRD$GB_CRD_DEBUG, 63$          ; 0734
         14 A0  DD 0092F     PUSHL   20(DDB_PTR)              ;
            50  DD 00932     PUSHL   DDB_PTR              ;
         00000000'   EF  9F 00934    PUSHAB  P.ABK              ;
```

```
            01  DD 0093A     PUSHL   #1              ;
            1A  DD 0093C     PUSHL   #26             ;
 00000000G  FF       05  FB 0093E    CALLS   #5, @CRD$PRINT          ;
            53       52  D0 00945 63$:   MOVL    R2, PREV_DDB_PTR        ; 0735
      04    AE       2C  D0 00948    MOVL    #44, SIZE               ; 0741
            5E  DD 0094C     PUSHL   SP              ;
      08    AE  9F 0094E     PUSHAB  SIZE            ;
 00000000V  EF       02  FB 00951    CALLS   #2, CRD$NEW             ;
            0C       50  E8 00958    BLBS    R0, 64$                 ;
            7E  7C 0095B     CLRQ    -(SP)           ;
            7E       01  CE 0095D    MNEGL   #1, -(SP)               ;
 00000000G  EF       03  FB 00960    CALLS   #3, CRD$INTERNAL_ERROR          ;
            13 00000000G  EF  E9 00967 64$:   BLBC    CRD$GB_CRD_DEBUG, 65$           ;
            6E  DD 0096E     PUSHL   DDB_PTR         ;
 00000000'  EF  9F 00970     PUSHAB  P.ABL           ;
            01  DD 00976     PUSHL   #1              ;
            1A  DD 00978     PUSHL   #26             ;
 00000000G  FF       04  FB 0097A    CALLS   #4, @CRD$PRINT          ;
            63 00    BE  0E 00981 65$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0743
            52       6E  D0 00985    MOVL    DDB_PTR, R2             ; 0745
      0C    A2 00000000G  EF  9E 00988    MOVAB   CRD$AL_UDA_2_HDWR_SPRT_TABLE+64, 12(R2)     ;
      10    A2  D4 00990    CLRL    16(R2)          ; 0746
      08    A2  D4 00993    CLRL    8(R2)           ; 0747
            52  DD 00996     PUSHL   R2              ; 0749
 00000000V  EF       01  FB 00998    CALLS   #1, CRD$INIT_DDB_STATUS         ;
      17    A2 04    AE  90 0099F    MOVB    SIZE, 23(R2)            ; 0752
            16 00000000G  EF  E9 009A4    BLBC    CRD$GB_CRD_DEBUG, 66$           ; 0755
      14    A0  DD 009AB     PUSHL   20(DDB_PTR)             ;
            50  DD 009AE     PUSHL   DDB_PTR         ;
 00000000'  EF  9F 009B0     PUSHAB  P.ABM           ;
            01  DD 009B6     PUSHL   #1              ;
            1A  DD 009B8     PUSHL   #26             ;
 00000000G  FF       05  FB 009BA    CALLS   #5, @CRD$PRINT          ;
            53       52  D0 009C1 66$:   MOVL    R2, PREV_DDB_PTR        ; 0756
      04    AE       2C  D0 009C4    MOVL    #44, SIZE               ; 0762
            5E  DD 009C8     PUSHL   SP              ;
      08    AE  9F 009CA     PUSHAB  SIZE            ;
 00000000V  EF       02  FB 009CD    CALLS   #2, CRD$NEW             ;
            0C       50  E8 009D4    BLBS    R0, 67$                 ;
            7E  7C 009D7     CLRQ    -(SP)           ;
            7E       01  CE 009D9    MNEGL   #1, -(SP)               ;
 00000000G  EF       03  FB 009DC    CALLS   #3, CRD$INTERNAL_ERROR          ;
            13 00000000G  EF  E9 009E3 67$:   BLBC    CRD$GB_CRD_DEBUG, 68$           ;
            6E  DD 009EA     PUSHL   DDB_PTR         ;
 00000000'  EF  9F 009EC     PUSHAB  P.ABN           ;
            01  DD 009F2     PUSHL   #1              ;
            1A  DD 009F4     PUSHL   #26             ;
 00000000G  FF       04  FB 009F6    CALLS   #4, @CRD$PRINT          ;
            63 00    BE  0E 009FD 68$:   INSQUE  @DDB_PTR, (PREV_DDB_PTR)        ; 0764
            52       6E  D0 00A01    MOVL    DDB_PTR, R2             ; 0766
      0C    A2 00000000G  EF  9E 00A04    MOVAB   CRD$AL_UDA_2_HDWR_SPRT_TABLE+96, 12(R2)     ;
      10    A2  D4 00A0C     CLRL    16(R2)          ; 0767
      08    A2  D4 00A0F     CLRL    8(R2)           ; 0768
            52  DD 00A12     PUSHL   R2              ; 0770
 00000000V  EF       01  FB 00A14    CALLS   #1, CRD$INIT_DDB STATUS         ;
```

```
      17   A2 04   AE  90 00A1B     MOVB    SIZE, 23(R2)          ; 0772
           01 00000000G  EF  E8 00A20    BLBS    CRD$GB_CRD_DEBUG, 69$         ; 0775
        04 00A27    RET                 ;
      14   A0  DD 00A28 69$:   PUSHL   20(DDB_PTR)          ;
           50  DD 00A2B    PUSHL   DDB_PTR          ;
       00000000'  EF  9F 00A2D    PUSHAB  P.ABO          ;
         027D  31 00A33    BRW     86$          ;
           06      52  91 00A36 70$:   CMPB    R2, #6          ; 0783
           01  13 00A39    BEQL    71$          ;
        04 00A3B    RET                 ;
      04   AE      2C  D0 00A3C 71$:   MOVL    #44, SIZE          ; 0789
           5E  DD 00A40    PUSHL   SP          ;
      08   AE  9F 00A42    PUSHAB  SIZE          ;
       00000000V  EF       02  FB 00A45    CALLS   #2, CRD$NEW          ;
           0C      50  E8 00A4C    BLBS    R0, 72$          ;
           7E  7C 00A4F    CLRQ    -(SP)          ;
           7E      01  CE 00A51    MNEGL   #1, -(SP)          ;
       00000000G  EF       03  FB 00A54    CALLS   #3, CRD$INTERNAL_ERROR        ;
           13 00000000G  EF  E9 00A5B 72$:   BLBC    CRD$GB_CRD_DEBUG, 73$         ;
           6E  DD 00A62    PUSHL   DDB_PTR          ;
       00000000'  EF  9F 00A64    PUSHAB  P.ABP          ;
           01  DD 00A6A    PUSHL   #1          ;
           1A  DD 00A6C    PUSHL   #26          ;
       00000000G  FF       04  FB 00A6E    CALLS   #4, @CRD$PRINT          ;
           52      6E  D0 00A75 73$:   MOVL    DDB_PTR, R2          ; 0791
           62      52  D0 00A78    MOVL    R2, (R2)          ;
      04   A2      52  D0 00A7B    MOVL    R2, 4(R2)          ; 0792
      0C   A2 00000000G  EF  9E 00A7F    MOVAB   CRD$AL_UDA_3_HDWR_SPRT_TABLE, 12(R2)       ; 0794
      10   A2  D4 00A87    CLRL    16(R2)          ; 0795
      08   A2  D4 00A8A    CLRL    8(R2)          ; 0796
           52  DD 00A8D    PUSHL   R2          ; 0798
       00000000V  EF       01  FB 00A8F    CALLS   #1, CRD$INIT_DDB_STATUS        ;
      14   A2      0C  88 00A96    BISB2   #12, 20(R2)          ; 0803
      17   A2 04   AE  90 00A9A    MOVB    SIZE, 23(R2)          ; 0808
           16 00000000G  EF  E9 00A9F    BLBC    CRD$GB_CRD_DEBUG, 74$         ; 0811
      14   A0  DD 00AA6    PUSHL   20(DDB_PTR)          ;
           50  DD 00AA9    PUSHL   DDB_PTR          ;
       00000000'  EF  9F 00AAB    PUSHAB  P.ABQ          ;
           01  DD 00AB1    PUSHL   #1          ;
           1A  DD 00AB3    PUSHL   #26          ;
       00000000G  FF       05  FB 00AB5    CALLS   #5, @CRD$PRINT          ;
           53      52  D0 00ABC 74$:   MOVL    R2, PREV_DDB_PTR          ; 0812
       00000000'  EF       52  D0 00ABF    MOVL    R2, CRD$GA_FIRST_DIAGNOSTIC        ; 0814
       00000000'  EF       52  D0 00AC6    MOVL    R2, CRD$GA_CURRENT_DIAGNOSTIC        ; 0817
      04   AE      2C  D0 00ACD    MOVL    #44, SIZE          ; 0824
           5E  DD 00AD1    PUSHL   SP          ;
      08   AE  9F 00AD3    PUSHAB  SIZE          ;
       00000000V  EF       02  FB 00AD6    CALLS   #2, CRD$NEW          ;
           0C      50  E8 00ADD    BLBS    R0, 75$          ;
           7E  7C 00AE0    CLRQ    -(SP)          ;
           7E      01  CE 00AE2    MNEGL   #1, -(SP)          ;
       00000000G  EF       03  FB 00AE5    CALLS   #3, CRD$INTERNAL_ERROR        ;
           13 00000000G  EF  E9 00AEC 75$:   BLBC    CRD$GB_CRD_DEBUG, 76$         ;
           6E  DD 00AF3    PUSHL   DDB_PTR          ;
       00000000'  EF  9F 00AF5    PUSHAB  P.ABR          ;
```

```
                01  DD 00AFB     PUSHL   #1              ;
                1A  DD 00AFD     PUSHL   #26             ;
 00000000G      FF       04  FB 00AFF    CALLS   #4, aCRD$PRINT          ;
                63 00    BE  0E 00B06 76$:   INSQUE  aDDB_PTR, (PREV_DDB_PTR)        ; 0826
                52       6E  D0 00B0A    MOVL    DDB_PTR, R2            ; 0828
            0C  A2 00000000G  EF  9E 00B0D    MOVAB   CRD$AL_UDA_3_HDWR_SPRT_TABLE+32, 12(R2)     ;
            10  A2  D4 00B15    CLRL    16(R2)             ; 0829
            08  A2  D4 00B18    CLRL    8(R2)              ; 0830
                52  DD 00B1B    PUSHL   R2              ; 0832
 00000000V      EF       01  FB 00B1D    CALLS   #1, CRD$INIT_DDB_STATUS         ;
            14  A2       04  88 00B24    BISB2   #4, 20(R2)             ; 0834
            17  A2 04    AE  90 00B28    MOVB    SIZE, 23(R2)           ; 0837
                16 00000000G  EF  E9 00B2D    BLBC    CRD$GB_CRD_DEBUG, 77$          ; 0839
            14  A0  DD 00B34    PUSHL   20(DDB_PTR)            ;
                50  DD 00B37    PUSHL   DDB_PTR             ;
 00000000'      EF  9F 00B39    PUSHAB  P.ABS               ;
                01  DD 00B3F    PUSHL   #1              ;
                1A  DD 00B41    PUSHL   #26             ;
 00000000G      FF       05  FB 00B43    CALLS   #5, aCRD$PRINT          ;
                53       52  D0 00B4A 77$:   MOVL    R2, PREV_DDB_PTR          ; 0840
            04  AE       2C  D0 00B4D    MOVL    #44, SIZE              ; 0846
                5E  DD 00B51    PUSHL   SP              ;
            08  AE  9F 00B53    PUSHAB  SIZE            ;
 00000000V      EF       02  FB 00B56    CALLS   #2, CRD$NEW            ;
                0C       50  E8 00B5D    BLBS    R0, 78$            ;
                7E  7C 00B60    CLRQ    -(SP)              ;
                7E       01  CE 00B62    MNEGL   #1, -(SP)              ;
 00000000G      EF       03  FB 00B65    CALLS   #3, CRD$INTERNAL_ERROR          ;
                13 00000000G  EF  E9 00B6C 78$:   BLBC    CRD$GB_CRD_DEBUG, 79$          ;
                6E  DD 00B73    PUSHL   DDB_PTR             ;
 00000000'      EF  9F 00B75    PUSHAB  P.ABT               ;
                01  DD 00B7B    PUSHL   #1              ;
                1A  DD 00B7D    PUSHL   #26             ;
 00000000G      FF       04  FB 00B7F    CALLS   #4, aCRD$PRINT          ;
                63 00    BE  0E 00B86 79$:   INSQUE  aDDB_PTR, (PREV_DDB_PTR)        ; 0848
                52       6E  D0 00B8A    MOVL    DDB_PTR, R2            ; 0850
            0C  A2 00000000G  EF  9E 00B8D    MOVAB   CRD$AL_UDA_3_HDWR_SPRT_TABLE+64, 12(R2)     ;
            10  A2  D4 00B95    CLRL    16(R2)             ; 0851
            08  A2  D4 00B98    CLRL    8(R2)              ; 0852
                52  DD 00B9B    PUSHL   R2              ; 0854
 00000000V      EF       01  FB 00B9D    CALLS   #1, CRD$INIT_DDB_STATUS         ;
            17  A2 04    AE  90 00BA4    MOVB    SIZE, 23(R2)           ; 0856
                16 00000000G  EF  E9 00BA9    BLBC    CRD$GB_CRD_DEBUG, 80$          ; 0859
            14  A0  DD 00BB0    PUSHL   20(DDB_PTR)            ;
                50  DD 00BB3    PUSHL   DDB_PTR             ;
 00000000'      EF  9F 00BB5    PUSHAB  P.ABU               ;
                01  DD 00BBB    PUSHL   #1              ;
                1A  DD 00BBD    PUSHL   #26             ;
 00000000G      FF       05  FB 00BBF    CALLS   #5, aCRD$PRINT          ;
                53       52  D0 00BC6 80$:   MOVL    R2, PREV_DDB_PTR          ; 0860
            04  AE       2C  D0 00BC9    MOVL    #44, SIZE              ; 0866
                5E  DD 00BCD    PUSHL   SP              ;
            08  AE  9F 00BCF    PUSHAB  SIZE            ;
 00000000V      EF       02  FB 00BD2    CALLS   #2, CRD$NEW            ;
                0C       50  E8 00BD9    BLBS    R0, 81$            ;
```

```
                7E  7C 00BDC     CLRQ    -(SP)                   ;
                7E     01  CE 00BDE     MNEGL   #1, -(SP)               ;
         00000000G  EF     03  FB 00BE1     CALLS   #3, CRD$INTERNAL_ERROR          ;
                13 00000000G  EF  E9 00BE8 81$:   BLBC    CRD$GB_CRD_DEBUG, 82$           ;
                6E  DD 00BEF     PUSHL   DDB_PTR                 ;
         00000000'  EF  9F 00BF1     PUSHAB  P.ABV                   ;
                01  DD 00BF7     PUSHL   #1                      ;
                1A  DD 00BF9     PUSHL   #26                     ;
         00000000G  FF     04  FB 00BFB     CALLS   #4, aCRD$PRINT          ;
                63 00   BE  0E 00C02 82$:   INSQUE  aDDB_PTR, (PREV_DDB_PTR)        ; 0868
                52     6E  D0 00C06     MOVL    DDB_PTR, R2             ; 0870
            0C  A2 00000000G  EF  9E 00C09     MOVAB   CRD$AL_UDA_3_HDWR_SPRT_TABLE+96, 12(R2)     ;
            10  A2  D4 00C11     CLRL    16(R2)                  ; 0871
            08  A2  D4 00C14     CLRL    8(R2)                   ; 0872
                52  DD 00C17     PUSHL   R2                      ; 0874
         00000000V  EF     01  FB 00C19     CALLS   #1, CRD$INIT_DDB_STATUS         ;
            17  A2 04   AE  90 00C20     MOVB    SIZE, 23(R2)            ; 0877
                16 00000000G  EF  E9 00C25     BLBC    CRD$GB_CRD_DEBUG, 83$           ; 0880
            14  A0  DD 00C2C     PUSHL   20(DDB_PTR)             ;
                50  DD 00C2F     PUSHL   DDB_PTR                 ;
         00000000'  EF  9F 00C31     PUSHAB  P.ABW                   ;
                01  DD 00C37     PUSHL   #1                      ;
                1A  DD 00C39     PUSHL   #26                     ;
         00000000G  FF     05  FB 00C3B     CALLS   #5, aCRD$PRINT          ;
                53     52  D0 00C42 83$:   MOVL    R2, PREV_DDB_PTR            ; 0881
            04  AE     2C  D0 00C45     MOVL    #44, SIZE               ; 0887
                5E  DD 00C49     PUSHL   SP                      ;
            08  AE  9F 00C4B     PUSHAB  SIZE                    ;
         00000000V  EF     02  FB 00C4E     CALLS   #2, CRD$NEW             ;
                0C     50  E8 00C55     BLBS    R0, 84$                 ;
                7E  7C 00C58     CLRQ    -(SP)                   ;
                7E     01  CE 00C5A     MNEGL   #1, -(SP)               ;
         00000000G  EF     03  FB 00C5D     CALLS   #3, CRD$INTERNAL_ERROR          ;
                13 00000000G  EF  E9 00C64 84$:   BLBC    CRD$GB_CRD_DEBUG, 85$           ;
                6E  DD 00C6B     PUSHL   DDB_PTR                 ;
         00000000'  EF  9F 00C6D     PUSHAB  P.ABX                   ;
                01  DD 00C73     PUSHL   #1                      ;
                1A  DD 00C75     PUSHL   #26                     ;
         00000000G  FF     04  FB 00C77     CALLS   #4, aCRD$PRINT          ;
                63 00   BE  0E 00C7E 85$:   INSQUE  aDDB_PTR, (PREV_DDB_PTR)        ; 0889
                52     6E  D0 00C82     MOVL    DDB_PTR, R2             ; 0891
            0C  A2 00000000G  EF  9E 00C85     MOVAB   CRD$AL_UDA_3_HDWR_SPRT_TABLE+128, 12(R2)    ;
            10  A2  D4 00C8D     CLRL    16(R2)                  ; 0892
            08  A2  D4 00C90     CLRL    8(R2)                   ; 0893
                52  DD 00C93     PUSHL   R2                      ; 0895
         00000000V  EF     01  FB 00C95     CALLS   #1, CRD$INIT_DDB_STATUS         ;
            17  A2 04   AE  90 00C9C     MOVB    SIZE, 23(R2)            ; 0897
                16 00000000G  EF  E9 00CA1     BLBC    CRD$GB_CRD_DEBUG, 87$           ; 0900
            14  A0  DD 00CA8     PUSHL   20(DDB_PTR)             ;
                50  DD 00CAB     PUSHL   DDB_PTR                 ;
         00000000'  EF  9F 00CAD     PUSHAB  P.ABY                   ;
                01  DD 00CB3 86$:   PUSHL   #1                      ;
                1A  DD 00CB5     PUSHL   #26                     ;
         00000000G  FF     05  FB 00CB7     CALLS   #5, aCRD$PRINT          ;
          04 00CBE 87$:   RET                     ; 0910
```

```
; Routine Size:  3263 bytes,    Routine Base:  CODE + 0000
```

```
;    0911  1 %SBTTL 'CRD$Dispose'                ![01]
;    0912  1                                     ![01]
;    0913  1 GLOBAL ROUTINE CRD$Dispose (Size, Address) =          ![01]
;    0914  1                                     ![01]
;    0915  1 !++
;    0916  1 ! Functional Description:
;    0917  1 !
;    0918  1 ! Deallocate a block of non-paged pool memory.
;    0919  1 !
;    0920  1 ! Formal Input Parameters:
;    0921  1 !
;    0922  1 ! Size    : The size of the block of memory to deallocate.
;    0923  1 ! Address : The address of the block of pool to deallocate.
;    0924  1 !
;    0925  1 ! Formal Output Parameters:
;    0926  1 !
;    0927  1 ! None
;    0928  1 !
;    0929  1 ! Side Effects:
;    0930  1 !
;    0931  1 ! None
;    0932  1 !
;    0933  1 ! Completion Codes:
;    0934  1 !
;    0935  1 ! None
;    0936  1 !-
```

```
;    0937  2 BEGIN     ! Routine CRD$Dispose       ![01]
;    0938  2  BIND                ![01]
;    0939  2   Block_To_Deallocate = (Address);        ![01]
;    0940  2
;    0941  2  MAP                 ![01]
;    0942  2   Address : REF BBLOCK;           ![01]
;    0943  2      ! Reference Address as a block of bytes     ![01]
;    0944  2
;    0945  2  BIND ROUTINE              ![01]
;    0946  2   CRD$DeAllocate_Memory = (.CRD$Exe$DeaNonPaged) : JSB_DeAll;     ![01]
;    0947  2      ! Use indirection to get the actual DS routine.    ![01]
;    0948  2
;    0949  2  IF (.CRD$GB_CRD_Debug) THEN          ![04]
; P  0950  2   $CRD_Print ($ASCIC ('Dispose::   Size: !8SL(D) = !8XL(X) Address: !8XL(X)!/'),   ![09]
;    0951  2    .Size, .Size, .Address);        ![04]
;    0952  2
;    0953  2  !+                  ![01]
;    0954  2  ! First, fill in the size and type (null is okay) of the block to be deallocated.   ![01]
;    0955  2  ! These fields are filled in in the block itself. Then, pass the address of the block   ![01]
;    0956  2  ! to the Exe$DeaNonPaged (Resident-DS) routine. Then return success. Note that the   ![01]
;    0957  2  ! Exe$DeaNonPaged routine will do a BugCheck if the deallocation is illegal.    ![01]
;    0958     .-                  ![01]
;    0959  2
;    0960  2  Address [IRP$W_Size] = .Size;          ![01]
;    0961  2  Address [IRP$B_Type] = 0;          ![01]
;    0962  2
;    0963  2  CRD$DeAllocate_Memory (.Block_To_Deallocate);        .[01]
;    0964  2  RETURN (CRD$K_Success);          ![01]
;    0965  1 END;     ! Routine CRD$Dispose       ![01]



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

69  53  20  20  20  3A  3A  65  73  6F  70  73  69  44  36  009E4 P.ABZ:  .ASCII  \6Dispose::   Size: !8SL(D) = !8XL(X) Add\  ;
21  20  3D  20  29  44  28  4C  53  38  21  20  3A  65  7A  009F3             ;
      64  64  41  20  29  58  28  4C  58  38  00A02                 ;
2F  21  29  58  28  4C  58  38  21  20  3A  73  73  65  72  00A0C    .ASCII  \ress: !8XL(X)!/\         ;

            .EXTRN  CRD$EXE$DEANONPAGED

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0FFC 00000    .ENTRY  CRD$DISPOSE, Save R2,R3,R4,R5,R6,R7,R8,R9,- ; 0913
             R10,R11          ;
            52 00000000G  EF  D0 00002    MOVL    CRD$EXE$DEANONPAGED, R2        ; 0946
            18 00000000G  EF  E9 00009    BLBC    CRD$GB_CRD_DEBUG, 1$          ; 0949
            7E 04   AC  7D 00010    MOVQ    SIZE, -(SP)           ; 0951
         04   AC  DD 00014    PUSHL   SIZE            ;
          00000000'  EF  9F 00017    PUSHAB  P.ABZ           ;
            01  DD 0001D    PUSHL   #1            ;
            1A  DD 0001F    PUSHL   #26           ;
          00000000G  FF      06  FB 00021    CALLS   #6, @CRD$PRINT          ;
            50 08   AC  D0 00028 1$:   MOVL    ADDRESS, R0           ; 0960
         08   A0 04   AC  B0 0002C    MOVW    SIZE, 8(R0)           ;
```

```
      0A    A0  94 00031    CLRB    10(R0)          ; 0961
            62  16 00034    JSB     (R2)          ; 0963
            50      01  D0 00036    MOVL    #1, R0          ; 0964
          04 00039    RET                 ; 0965

; Routine Size:  58 bytes,    Routine Base:  CODE + 0CBF
```

```
;    0966  1 %SBTTL 'CRD$Init_DDB_Status'
;    0967  1
;    0968  1 GLOBAL ROUTINE CRD$Init_DDB_Status (DDB_Ptr) : NOVALUE =
;    0969  1
;    0970  1 !++
;    0971  1 ! Functional Description:
;    0972  1 !
;    0973  1 ! This routine will initialize all the status bits in a DDB to their default values.
;    0974  1 ! This routine can be used by both Auto and Menu test.
;    0975  1 !
;    0976  1 ! Formal Input Parameters:
;    0977  1 !
;    0978  1 ! DDB_Ptr: A pointer to a Device_Data_Block_Structure
;    0979  1 !
;    0980  1 ! Formal Output Parameters:
;    0981  1 !
;    0982  1 ! None
;    0983  1 !
;    0984  1 ! Side Effects:
;    0985  1 !
;    0986  1 ! Initializes the bits in the status word.
;    0987  1 !
;    0988  1 ! Completion Codes:
;    0989  1 !
;    0990  1 ! None
;    0991  1 !--
```

```
;   0992  2 BEGIN     ! Routine CRD$Init_DDB_Status
;   0993  2  MAP
;   0994  2   DDB_Ptr : REF Device_Data_Block_Structure;
;   0995  2
;   0996  2  DDB_Ptr [CRD$K_DDB_Status] = 0;           ![07]
;   0997  2
;   0998  2  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete> = True;      ![07]
;   0999  2
;   1000  2  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Diagnostic_Error>  = False;      ![07]
;   1001  2  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Preparation_Error> = False;      ![07]
;   1002  2  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Device_Bad>        = False;      ![07]
;   1003  2  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Check_Device_Type> = False;      ![07]
;   1004  2  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device>  = False;      ![07]
;   1005  2  DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed>   = False;      ![07]
;   1006  1 END;     ! Routine CRD$Init_DDB_Status



                0000 00000     .ENTRY  CRD$INIT_DDB_STATUS, Save nothing       ; 0968
              50 04   AC  D0 00002     MOVL    DDB_PTR, R0           ; 0996
              50       14  C0 00006     ADDL2   #20, R0          ;
              60  D4 00009     CLRL    (R0)             ;
              60       20  88 0000B     BISB2   #32, (R0)           ; 0998
              60 5F    8F  8A 0000E     BICB2   #95, (R0)           ; 1005
           04 00012     RET                 ; 1006

; Routine Size:  19 bytes,    Routine Base:  CODE + 0CF9
```

```
;    1007  1 %SBTTL 'CRD$New'                ![01]
;    1008  1                                 ![01]
;    1009  1 GLOBAL ROUTINE CRD$New (Size, Address) =          ![01]
;    1010  1                                 ![01]
;    1011  1 !++
;    1012  1 ! Functional Description:
;    1013  1 !
;    1014  1 ! Allocate a block of non-paged pool memory.
;    1015  1 !
;    1016  1 ! Formal Input Parameters:
;    1017  1 !
;    1018  1 ! Size     : The address of the size (in BYTES) of the block of memory needed.    ![03]
;    1019  1 !
;    1020  1 ! Formal Output Parameters:
;    1021  1 !
;    1022  1 ! Address : The address of the allocated block of pool.
;    1023  1 !
;    1024  1 ! Side Effects:
;    1025  1 !
;    1026  1 ! None
;    1027  1 !
;    1028  1 ! Completion Codes:
;    1029  1 !
;    1030  1 ! CRD$K_Failure - No memory allocated
;    1031  1 ! CRD$K_Success - Memory size allocated
;    1032  1 !--
```

```
;   1033  2 BEGIN     ! Routine CRD$New       ![01]
;   1034  2  LOCAL              ![01]
;   1035  2   Returned_Size;  ! The size returned by the memory allocation routine.   ![01]
;   1036  2
;   1037  2  BIND ROUTINE             ![01]
;   1038  2   CRD$Allocate_Memory = (.CRD$Exe$AloNonPaged) : JSB_Alloc;     ![01]
;   1039  2      ! Use indirection to get the actual DS routine.    ![01]
;   1040  2
;   1041  2  !+               ![01]
;   1042  2  ! First, call the allocation routine. If successful, it will return a size of the block allocated, and ![01]
;   1043  2  ! the address of the block. Then check the size of the allocated block. Make sure it is big enough. ![01]
;   1044  2  ! If the allocation routine fails, or if the size is not big enough, return failure.   ![01]
;   1045  2  !-               ![01]
;   1046  2
;   1047  2  IF (CRD$Allocate_Memory (..Size; Returned_Size, .Address)) THEN      ![03]
;   1048  2   IF (.Returned_Size GEQ ..Size) THEN         ![03]
;   1049  3    BEGIN             ![03]
;   1050  3    IF (.CRD$GB_CRD_Debug) THEN         ![04]
; P 1051  3     $CRD_Print ($ASCIC          ![04]
; P 1052  3      ('New::       Size: !8SL(D) Address: !8XL(X) Return size: !8SL(D)!/'), ![09]
;   1053  3      ..Size, .Address, .Returned_Size);     ![04]
;   1054  3
;   1055  3    .Size = .Returned_Size;          ![03]
;   1056  4    RETURN (CRD$K_Success)          ![03]
;   1057  2    END;             ![03]
;   1058  2
;   1059  2  IF (.CRD$GB_CRD_Debug) THEN           ![04]
; P 1060  2   $CRD_Print ($ASCIC ('New::       Size: !8SL(D) Address: !8XL(X) Return size: !8SL(D)!/'), ![09]
;   1061  2    ..Size, .Address, .Returned_Size);       ![04]
;   1062  2
;   1063  2  RETURN (CRD$K_Failure);           ![01]
;   1064  1 END;     ! Routine CRD$New       ![01]



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

69  53  20  20  20  20  20  20  20  3A  3A  77  65  4E  41   00A1B P.ACA:  .ASCII  \ANew::       Size: !8SL(D) Address: !8XL\  ;
64  64  41  20  29  44  28  4C  53  38  21  20  3A  65  7A   00A2A              ;
      4C  58  38  21  20  3A  73  73  65  72  00A39                ;
65  7A  69  73  20  6E  72  75  74  65  52  20  29  58  28   00A43    .ASCII  \(X) Return size: !8SL(D)!/\        ;
  2F  21  29  44  28  4C  53  38  21  20  3A  00A52                ;
69  53  20  20  20  20  20  20  20  3A  3A  77  65  4E  41   00A5D P.ACB:  .ASCII  \ANew::       Size: !8SL(D) Address: !8XL\  ;
64  64  41  20  29  44  28  4C  53  38  21  20  3A  65  7A   00A6C              ;
      4C  58  38  21  20  3A  73  73  65  72  00A7B                ;
65  7A  69  73  20  6E  72  75  74  65  52  20  29  58  28   00A85    .ASCII  \(X) Return size: !8SL(D)./\        ;
  2F  21  29  44  28  4C  53  38  21  20  3A  00A94                ;

            .EXTRN  CRD$EXE$ALONONPAGED

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0FFC 00000    .ENTRY  CRD$NEW, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,- ; 1009
             R11             ;
            51 04   BC  D0 00002    MOVL    aSIZE, R1          ; 1047
```

```
                00000000G  FF  16 00006    JSB     @CRD$EXE$ALONONPAGED          ;
                    54      51  D0 0000C    MOVL    R1, R4              ;
                08  BC      52  D0 0000F    MOVL    R2, @ADDRESS            ;
                    2E      50  E9 00013    BLBC    R0, 2$              ;
                04  BC      54  D1 00016    CMPL    RETURNED_SIZE, @SIZE          ; 1048
                    28  19 0001A    BLSS    2$              ;
                    19 00000000G  EF  E9 0001C    BLBC    CRD$GB_CRD_DEBUG, 1$          ; 1050
                    54  DD 00023    PUSHL   RETURNED_SIZE           ; 1053
                08  AC  DD 00025    PUSHL   ADDRESS             ;
                04  BC  DD 00028    PUSHL   @SIZE               ;
                 00000000'  EF  9F 0002B    PUSHAB  P.ACA               ;
                    01  DD 00031    PUSHL   #1              ;
                    1A  DD 00033    PUSHL   #26             ;
                 00000000G  FF      06  FB 00035    CALLS   #6, @CRD$PRINT          ;
                04  BC      54  D0 0003C 1$:   MOVL    RETURNED_SIZE, @SIZE          ; 1055
                    50      01  D0 00040    MOVL    #1, R0              ; 1056
                  04 00043    RET                 ;
                    19 00000000G  EF  E9 00044 2$:   BLBC    CRD$GB_CRD_DEBUG, 3$          ; 1059
                    54  DD 0004B    PUSHL   RETURNED_SIZE           ; 1061
                08  AC  DD 0004D    PUSHL   ADDRESS             ;
                04  BC  DD 00050    PUSHL   @SIZE               ;
                 00000000'  EF  9F 00053    PUSHAB  P.ACB               ;
                    01  DD 00059    PUSHL   #1              ;
                    1A  DD 0005B    PUSHL   #26             ;
                 00000000G  FF      06  FB 0005D    CALLS   #6, @CRD$PRINT          ;
                    50  D4 00064 3$:   CLRL    R0              ; 1063
                  04 00066    RET                 ; 1064

; Routine Size:  103 bytes,    Routine Base:  CODE + 0D0C
```

```
;   1065  1 END     ! End of CRDDDB module
;   1066  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          2719  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK             8  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          3443  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$ASCIC    Unbound             190     205
    Macro    Lib02   230     252     265     280     287     300     307     323     330     343
             358     376     389     404     452     474     488     501     508     522
             529     542     556     578     591     606     613     626     633     647
             654     667     685     707     721     734     741     755     762     775
             789     811     824     839     846     859     866     880     887     900
             950    1051    1060
$CRD_LITERALS   Macro    Lib03   120
$CRD_PRINT   Unbound             190     205
    Macro    Lib03   230     252     265     280     287     300     307     323     330     343
             358     376     389     404     452     474     488     501     508     522
             529     542     556     578     591     606     613     626     633     647
             654     667     685     707     721     734     741     755     762     775
             789     811     824     839     846     859     866     880     887     900
             950    1051    1060
$DS_DSADEF   Macro    Lib02   119
$DS_TYPEDEF   Macro    Lib02   230     252     265     280     287     300     307     323     330     343
             358     376     389     404     452     474     488     501     508     522
             529     542     556     578     591     606     613     626     633     647
             654     667     685     707     721     734     741     755     762     775
             789     811     824     839     846     859     866     880     887     900
             951    1053    1061
$EQULST    Macro    Lib04   120     230     252     265     280     287     300     307     323     330
             343     358     376     389     404     452     474     488     501     508
             522     529     542     556     578     591     606     613     626     633
             647     654     667     685     707     721     734     741     755     762
             775     789     811     824     839     846     859     866     880     887
             900     951    1053    1061
$ER    Comptime   136
$MODULE    Bind        136
ADDRESS    RoutForm   913   939a    942m    951.    960a=   961a=
    RoutForm  1009  1047a=  1053.   1061.
ALLOCATE_DDB   Macro      197   230     265     287     307     330     358     389     452     488     508
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
              529     556     591     613     633     654     685     721     741     762
              789     824     846     866     887
AUTO$K_EXIT_AUTOSIZER_ERROR Literal    120
AUTO$K_EXIT_CONTROL_C  Literal    120
AUTO$K_EXIT_DUMMY_2  Literal    120
AUTO$K_EXIT_DUMMY_3  Literal    120
AUTO$K_EXIT_ERRORS_COMPLETE Literal    120
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal    120
AUTO$K_EXIT_INTERNAL_ERROR Literal    120
AUTO$K_EXIT_INV_BASE_SYSTEM Literal    120
AUTO$K_EXIT_LAST_REASON  Literal    120   120
AUTO$K_EXIT_NOERRORS_COMPLETE Literal    120
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal    120
AUTO$K_EXIT_NOERRORS_PREP Literal    120
BBLOCK    Structur Lib01   942
BLOCK_TO_DEALLOCATE  Bind       939   963.
CRD$ALLOCATE_MEMORY  BindRout  1038  1047c
CRD$AL_IDC_HDWR_SPRT_TABLE External Lib03   235a    269a    291a    311a    334a
CRD$AL_LESI_HDWR_SPRT_TABLE External Lib03   363a    393a
CRD$AL_UDA_0_HDWR_SPRT_TABLE External Lib03   457a    492a    512a    533a
CRD$AL_UDA_1_HDWR_SPRT_TABLE External Lib03   561a    595a    617a    637a    658a
CRD$AL_UDA_2_HDWR_SPRT_TABLE External Lib03   690a    725a    745a    766a
CRD$AL_UDA_3_HDWR_SPRT_TABLE External Lib03   794a    828a    850a    870a    891a
CRD$BUILD_DDBS   GlobRout   149 Lib03e    106f
CRD$DEALLOCATE_MEMORY  BindRout   946   963c
CRD$DISPOSE   GlobRout   913 Lib03e    109f
CRD$EXE$ALONONPAGED  External Lib03  1038.
CRD$EXE$DEANONPAGED  External Lib03   946.
CRD$GA_CURRENT_DIAGNOSTIC Global     140 Lib03e    258=    382=    480=    584=    713=    817=
CRD$GA_FIRST_DIAGNOSTIC  Global     142 Lib03e    142=    255=    379=    477=    581=    710=    814=
CRD$GB_BASE_SYSTEM  External Lib03   221.
CRD$GB_CRD_DEBUG  Unbound          184     204
    External Lib03   230.    252.    265.    280.    287.    300.    307.    323.    330.    343.
             358.    376.    389.    404.    452.    474.    488.    501.    508.    522.
             529.    542.    556.    578.    591.    606.    613.    626.    633.    647.
             654.    667.    685.    707.    721.    734.    741.    755.    762.    775.
             789.    811.    824.    839.    846.    859.    866.    880.    887.    900.
             949.   1050.   1059.
CRD$INIT_DDB_STATUS  GlobRout   968 Lib03e    112f    239c    273c    295c    315c    338c    367c    397c    461c
             496c    516c    537c    565c    599c    621c    641c    662c    694c    729c
             749c    770c    798c    832c    854c    874c    895c
CRD$INTERNAL_ERROR  Unbound          201
    ExtRout  Lib03   230c    265c    287c    307c    330c    358c    389c    452c    488c    508c
             529c    556c    591c    613c    633c    654c    685c    721c    741c    762c
             789c    824c    846c    866c    887c
CRD$K_ACTION_EQL_DO_NOTHING Literal    120
CRD$K_ACTION_RANGE_ERROR Literal    120
CRD$K_ADVANCE_DIAGNOSTIC Literal    120
CRD$K_ADVANCE_GENERAL_COM Literal    120
CRD$K_ADVANCE_STATE  Literal    120
CRD$K_ALLOCATION_ERROR  Unbound          201
    Literal    120   230     265     287     307     330     358     389     452     488     508
             529     556     591     613     633     654     685     721     741     762
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
              789     824     846     866     887
CRD$K_ASSIGN_PTABLE_PTRS Literal     120
CRD$K_AUTOSIZER_ERROR  Literal     120
CRD$K_AUTOSIZER_SET_FLAGS Literal     120
CRD$K_BAD_ACTION_NUMBER  Literal     120
CRD$K_BAD_TYPECODE_ERROR Literal     120
CRD$K_BASE_SYSTEM_IDC  Literal     120   223
CRD$K_BASE_SYSTEM_LESI  Literal     120   352
CRD$K_BASE_SYSTEM_NULL  Literal     120
CRD$K_BASE_SYSTEM_UDA_0  Literal     120   446
CRD$K_BASE_SYSTEM_UDA_1  Literal     120   550
CRD$K_BASE_SYSTEM_UDA_2  Literal     120   679
CRD$K_BASE_SYSTEM_UDA_3  Literal     120   783
CRD$K_CRD_INITIALIZATION Literal     120
CRD$K_CREATE_HST  Literal     120
CRD$K_DATA_LENGTH_ERROR  Literal     120
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     120   235     269     291     311     334     363     393     457     492     512
              533     561     595     617     637     658     690     725     745     766
              794     828     850     870     891
CRD$K_DDB_BLINK   Literal     120   233     361     455     559     688     792
CRD$K_DDB_FLINK   Literal     120   232     360     454     558     687     791
CRD$K_DDB_LAST_ENTRY  Literal     120
CRD$K_DDB_MEMORY_SIZE  Literal     120
CRD$K_DDB_MENU_DEVICE_NUMB Literal     120
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     120   236     270     292     312     335     364     394     458     493     513
              534     562     596     618     638     659     691     726     746     767
              795     829     851     871     892
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     120
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     120
CRD$K_DDB_PTABLE_ENTRY  Literal     120   237     271     293     313     336     365     395     459     494     514
              535     563     597     619     639     660     692     727     747     768
              796     830     852     872     893
CRD$K_DDB_SIZE   Unbound          198
    Literal  Lib03   230     265     287     307     330     358     389     452     488     508
              529     556     591     613     633     654     685     721     741     762
              789     824     846     866     887
CRD$K_DDB_STATUS  Unbound          192
    Literal     120   241     244     249     252     275     278     280     297     300     317
              320     323     340     343     369     373     376     399     402     404
              463     466     471     474     498     501     519     522     539     542
              567     570     575     578     601     604     606     623     626     644
              647     664     667     696     699     704     707     731     734     752
              755     772     775     800     803     808     811     834     837     839
              856     859     877     880     897     900     996     998    1000    1001
             1002    1003    1004    1005
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     120
CRD$K_DEVICE_NAME  Literal     120
CRD$K_DIAGNOSTIC_ERROR  Literal     120
CRD$K_DIAGNOSTIC_NAME  Literal     120
CRD$K_DONE_GENERAL_COMS  Literal     120
CRD$K_DO_NOTHING  Literal     120
CRD$K_DUMMY_10   Literal     120
CRD$K_DUMMY_11   Literal     120
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_DUMMY_12   Literal    120
CRD$K_DUMMY_13   Literal    120
CRD$K_DUMMY_3   Literal    120
CRD$K_DUMMY_4   Literal    120
CRD$K_DUMMY_5   Literal    120
CRD$K_DUMMY_6   Literal    120
CRD$K_DUMMY_7   Literal    120
CRD$K_DUMMY_8   Literal    120
CRD$K_DUMMY_9   Literal    120
CRD$K_EXIT_CRD   Literal    120
CRD$K_FAILURE   Literal  Lib03  1063
CRD$K_HOOK_POINT  Literal    120
CRD$K_HS_INTERNAL_ERROR  Literal    120
CRD$K_IDC_EVDLB   Literal    120   291
CRD$K_IDC_EVDLC   Literal    120   311
CRD$K_IDC_EVDLD   Literal    120   334
CRD$K_IDC_EVRAD_0  Literal    120   235
CRD$K_IDC_EVRAD_1  Literal    120   269
CRD$K_IDC_LAST_DIAGNOSTIC Literal    120
CRD$K_INTERNAL_ERROR  Literal    120
CRD$K_LAST_ACTION_ROUTINE Literal    120
CRD$K_LAST_ENTRY  Literal    120
CRD$K_LAST_FUNCTION  Literal    120
CRD$K_LAST_HOOK_POINT  Literal    120
CRD$K_LAST_INTERNAL_ERROR Literal    120
CRD$K_LAST_TYPE   Literal    120
CRD$K_LESI_ENWCA  Literal    120
CRD$K_LESI_EVRMB_0  Literal    120   363
CRD$K_LESI_EVRMB_1  Literal    120   393
CRD$K_LESI_LAST_DIAGNOSTIC Literal    120
CRD$K_LEVEL_4_PASSED  Literal    120
CRD$K_LOAD_AUTOSIZER  Literal    120
CRD$K_LOAD_ERROR  Literal    120
CRD$K_LOAD_GENERAL_COM  Literal    120
CRD$K_LOAD_LOAD_COM  Literal    120
CRD$K_LOAD_SELECT_COM  Literal    120
CRD$K_LOAD_SET_EVENT_COM Literal    120
CRD$K_LOAD_SET_FLAGS_COM Literal    120
CRD$K_LOAD_START_COM  Literal    120
CRD$K_LOAD_SUP_FILE  Literal    120
CRD$K_MM_INTERNAL_ERROR  Literal    120
CRD$K_NEW_LINE   Literal    120
CRD$K_PREPARATION_ERROR  Literal    120
CRD$K_PREPARATION_ERROR_TEXT Literal    120
CRD$K_RUNTIME   Literal    120
CRD$K_SAME_LINE   Literal    120
CRD$K_SET_EVENT_ARGS  Literal    120
CRD$K_SET_FLAGS_ARGS  Literal    120
CRD$K_SET_UP DIAGNOSTIC  Literal    120
CRD$K_START   Literal    120
CRD$K START_AUTOSIZER  Literal    120
CRD$K START_ERROR  Literal    120
CRD$K_START_QUALIFIERS  Literal    120
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------- -------- ----- ----------------
CRD$K_STAR_LINE   Literal    120
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal    120
CRD$K_STATE_ADVANCE_GENERAL_COM Literal    120
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal    120
CRD$K_STATE_AUTOSIZER_ERROR Literal    120
CRD$K_STATE_AUTOSIZER_RUNNING Literal    120
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal    120
CRD$K_STATE_DIAGNOSTIC_ERROR Literal    120
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal    120
CRD$K_STATE_DONE_DIAGNOSTICS Literal    120
CRD$K_STATE_DONE_GENERAL_COMS Literal    120
CRD$K_STATE_DUMMY_1  Literal    120
CRD$K_STATE_DUMMY_10  Literal    120
CRD$K_STATE_DUMMY_12  Literal    120
CRD$K_STATE_DUMMY_13  Literal    120
CRD$K_STATE_DUMMY_2  Literal    120
CRD$K_STATE_DUMMY_3  Literal    120
CRD$K_STATE_DUMMY_4  Literal    120
CRD$K_STATE_DUMMY_6  Literal    120
CRD$K_STATE_DUMMY_7  Literal    120
CRD$K_STATE_DUMMY_8  Literal    120
CRD$K_STATE_EXIT_CRD  Literal    120
CRD$K_STATE_INTERNAL_ERROR Literal    120
CRD$K_STATE_LAST_STATE  Literal    120
CRD$K_STATE_LEVEL_4_PASSED Literal    120
CRD$K_STATE_LOAD_AUTOSIZER Literal    120
CRD$K_STATE_LOAD_ERROR  Literal    120
CRD$K_STATE_LOAD_GENERAL_COM Literal    120
CRD$K_STATE_LOAD_LOAD_COM Literal    120
CRD$K_STATE_LOAD_SELECT_COM Literal    120
CRD$K_STATE_LOAD_SET_EVENT_COM Literal    120
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal    120
CRD$K_STATE_LOAD_START_COM Literal    120
CRD$K_STATE_PREPARATION_ERROR Literal    120
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal    120
CRD$K_STATE_START  Literal    120
CRD$K_STATE_START_AUTOSIZER Literal    120
CRD$K_STATE_START_ERROR  Literal    120
CRD$K_SUCCESS   Literal  Lib03   964    1056
CRD$K_TEST_START_TEXT  Literal    120
CRD$K_TQ_INTERNAL_ERROR  Literal    120
CRD$K_TYPECODE   Literal    120
CRD$K_UDA_0_EVDLB  Literal    120   492
CRD$K_UDA_0_EVDLC  Literal    120   512
CRD$K_UDA_0_EVDLD  Literal    120   533
CRD$K_UDA_0_EVRLA_0  Literal    120   457
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal    120
CRD$K_UDA_1_EVDLB  Literal    120   617
CRD$K_UDA_1_EVDLC  Literal    120   637
CRD$K_UDA_1_EVDLD  Literal    120   658
CRD$K_UDA_1_EVRLA_0  Literal    120   561
CRD$K_UDA_1_EVRLA_1  Literal    120   595
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal    120
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_UDA_2_EVDLB  Literal     120   725
CRD$K_UDA_2_EVDLC  Literal     120   745
CRD$K_UDA_2_EVDLD  Literal     120   766
CRD$K_UDA_2_EVRLA_0  Literal     120   690
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     120
CRD$K_UDA_3_EVDLB  Literal     120   850
CRD$K_UDA_3_EVDLC  Literal     120   870
CRD$K_UDA_3_EVDLD  Literal     120   891
CRD$K_UDA_3_EVRLA_0  Literal     120   794
CRD$K_UDA_3_EVRLA_1  Literal     120   828
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     120
CRD$NEW    Unbound          199
    GlobRout  1009 Lib03e    115f    230c    265c    287c    307c    330c    358c    389c    452c
              488c    508c    529c    556c    591c    613c    633c    654c    685c    721c
              741c    762c    789c    824c    846c    866c    887c
CRD$PRINT   External Lib03   230ac   252ac   265ac   280ac   287ac   300ac   307ac   323ac   330ac   343ac
              358ac   376ac   389ac   404ac   452ac   474ac   488ac   501ac   508ac   522ac
              529ac   542ac   556ac   578ac   591ac   606ac   613ac   626ac   633ac   647ac
              654ac   667ac   685ac   707ac   721ac   734ac   741ac   755ac   762ac   775ac
              789ac   811ac   824ac   839ac   846ac   859ac   866ac   880ac   887ac   900ac
              951ac  1053ac  1061ac
DDB    MacrForm   183   189
DDB$B_STATUS_DDB_SIZE  Macro    Lib03   249     278     297     320     340     373     402     471     498     519
              539     575     604     623     644     664     704     731     752     772
              808     837     856     877     897
DDB$V_STATUS_CHECK_DEVICE_TYPE Macro    Lib03   244     466     570     699     803     1003
DDB$V_STATUS_DEVICE_BAD  Macro    Lib03  1002
DDB$V_STATUS_DEV_TEST_COMPLETE Macro    Lib03   998
DDB$V_STATUS_DIAGNOSTIC_ERROR Macro    Lib03  1000
DDB$V_STATUS_ESSENTIAL_DEVICE Macro    Lib03   241     275     317     369     399     463     567     601     696     800
              834    1004
DDB$V_STATUS_MESSAGE_PRINTED Macro    Lib03  1005
DDB$V_STATUS_PREPARATION_ERROR Macro    Lib03  1001
DDB_PTR    Unbound          187     189     192     199     205
    Local      211   230a    230.    232a=   232.    233a=   233.    235a=   236a=   237a=   239.
              241a=   244a=   249a=   253.    255.    258.    265a    265.    267.    269a=
              270a=   271a=   273.    275a=   278a=   281.    287a    287.    289.    291a=
              292a=   293a=   295.    297a=   301.    307a    307.    309.    311a=   312a=
              313a=   315.    317a=   320a=   324.    330a    330.    332.    334a=   335a=
              336a=   338.    340a=   358a    358.    360a=   360.    361a=   361.    363a=
              364a=   365a=   367.    369a=   373a=   377.    379.    382.    389a    389.
              391.    393a=   394a=   395a=   397.    399a=   402a=   405.    452a    452.
              454a=   454.    455a=   455.    457a=   458a=   459a=   461.    463a=   466a=
              471a=   475.    477.    480.    488a    488.    490.    492a=   493a=   494a=
              496.    498a=   502.    508a    508.    510.    512a=   513a=   514a=   516.
              519a=   523.    529a    529.    531.    533a=   534a=   535a=   537.    539a=
              556a    556.    558a=   558.    559a=   559.    561a=   562a=   563a=   565.
              567a=   570a=   575a=   579.    581.    584.    591a    591.    593.    595a=
              596a=   597a=   599.    601a=   604a=   607.    613a    613.    615.    617a=
              618a=   619a=   621.    623a=   627.    633a    633.    635.    637a=   638a=
              639a=   641.    644a=   648.    654a    654.    656.    658a=   659a=   660a=
              662.    664a=   685a    685.    687a=   687.    688a=   688.    690a=   691a=
              692a=   694.    696a=   699a=   704a=   708.    710.    713.    721a    721.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
                723.    725a=   726a=   727a=   729.    731a=   735.    741a    741.    743.
                745a=   746a=   747a=   749.    752a=   756.    762a    762.    764.    766a=
                767a=   768a=   770.    772a=   789a    789.    791a=   791.    792a=   792.
                794a=   795a=   796a=   798.    800a=   803a=   808a=   812.    814.    817.
                824a    824.    826.    828a=   829a=   830a=   832.    834a=   837a=   840.
                846a    846.    848.    850a=   851a=   852a=   854.    856a=   860.    866a
                866.    868.    870a=   871a=   872a=   874.    877a=   881.    887a    887.
                889.    891a=   892a=   893a=   895.    897a=
    Local        252    252=     252.    252a.
    Local        280    280=     280.    280a.
    Local        300    300=     300.    300a.
    Local        323    323=     323.    323a.
    Local        343    343=     343.    343a.
    Local        376    376=     376.    376a.
    Local        404    404=     404.    404a.
    Local        474    474=     474.    474a.
    Local        501    501=     501.    501a.
    Local        522    522=     522.    522a.
    Local        542    542=     542.    542a.
    Local        578    578=     578.    578a.
    Local        606    606=     606.    606a.
    Local        626    626=     626.    626a.
    Local        647    647=     647.    647a.
    Local        667    667=     667.    667a.
    Local        707    707=     707.    707a.
    Local        734    734=     734.    734a.
    Local        755    755=     755.    755a.
    Local        775    775=     775.    775a.
    Local        811    811=     811.    811a.
    Local        839    839=     839.    839a.
    Local        859    859=     859.    859a.
    Local        880    880=     880.    880a.
    Local        900    900=     900.    900a.
    RoutForm     968    994m     996a=   998a=  1000a=  1001a=  1002a=  1003a=  1004a=  1005a=
DEVICE_DATA_BLOCK_STRUCTURE Unbound             187
    Structur Lib03   211     252      280      300      323      343      376      404      474      501
                     522      542      578      606      626      647      667      707      734      755
                     775      811      839      859      880      900      994
DIAG_STRING     MacrForm    197     205
DS$K_PRINTB     Literal     230
    Literal      252
    Literal      265
    Literal      280
    Literal      287
    Literal      300
    Literal      307
    Literal      323
    Literal      330
    Literal      343
    Literal      358
    Literal      376
    Literal      389
    Literal      404
```

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_PRINTF	Literal	230	230
	Literal	252	252
	Literal	265	265
	Literal	280	280
	Literal	287	287
	Literal	300	300
	Literal	307	307
	Literal	323	323
	Literal	330	330
	Literal	343	343
	Literal	358	358
	Literal	376	376
	Literal	389	389
	Literal	404	404

Symbol	Type	Defined	Referenced ...
	Literal	452	452
	Literal	474	474
	Literal	488	488
	Literal	501	501
	Literal	508	508
	Literal	522	522
	Literal	529	529
	Literal	542	542
	Literal	556	556
	Literal	578	578
	Literal	591	591
	Literal	606	606
	Literal	613	613
	Literal	626	626
	Literal	633	633
	Literal	647	647
	Literal	654	654
	Literal	667	667
	Literal	685	685
	Literal	707	707
	Literal	721	721
	Literal	734	734
	Literal	741	741
	Literal	755	755
	Literal	762	762
	Literal	775	775
	Literal	789	789
	Literal	811	811
	Literal	824	824
	Literal	839	839
	Literal	846	846
	Literal	859	859
	Literal	866	866
	Literal	880	880
	Literal	887	887
	Literal	900	900
	Literal	951	951
	Literal	1053	1053
	Literal	1061	1061
DS$K_PRINTI	Literal		230
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_PRINTX	Literal	230	
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_ABORT_PROGRAM  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_ABORT_TEST  Literal    230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_COMMAND_ERR  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_TYPE_COMMAND_OUT	Literal	230	
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_TYPE_CRD_AUTOTEST	Literal	230	230
	Literal	252	252
	Literal	265	265
	Literal	280	280
	Literal	287	287
	Literal	300	300
	Literal	307	307
	Literal	323	323
	Literal	330	330
	Literal	343	343
	Literal	358	358
	Literal	376	376
	Literal	389	389
	Literal	404	404

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
    Literal     452   452
    Literal     474   474
    Literal     488   488
    Literal     501   501
    Literal     508   508
    Literal     522   522
    Literal     529   529
    Literal     542   542
    Literal     556   556
    Literal     578   578
    Literal     591   591
    Literal     606   606
    Literal     613   613
    Literal     626   626
    Literal     633   633
    Literal     647   647
    Literal     654   654
    Literal     667   667
    Literal     685   685
    Literal     707   707
    Literal     721   721
    Literal     734   734
    Literal     741   741
    Literal     755   755
    Literal     762   762
    Literal     775   775
    Literal     789   789
    Literal     811   811
    Literal     824   824
    Literal     839   839
    Literal     846   846
    Literal     859   859
    Literal     866   866
    Literal     880   880
    Literal     887   887
    Literal     900   900
    Literal     951   951
    Literal    1053  1053
    Literal    1061  1061
DS$K_TYPE_DS_PROMPT  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_DS_START  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_ERRDEV	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_ERRHARD	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_ERROR_BODY	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_TYPE_ERROR_END	Literal	230	
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_ERRPREP  Literal    230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_ERRSOFT	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_ERRSUP	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_ERRSYS  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -- -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_ERR_HALT  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_EXCEPTION  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_EXCEPTION_HEAD Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_FIRST_PASS	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_TYPE_GENERAL	Literal	230	
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_GENERAL_ERROR	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_NO_TESTS	Literal	230	
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_PARAM_ERROR	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_PROGRAM_END  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_TYPE_PROGRAM_INFO	Literal	230	
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_PROGRAM_START	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_QIO_INVADP	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_QIO_NODRIVER	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_TYPE_QIO_WRONGVER	Literal	230	
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_SCRIPT_ECHO	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_SCRIPT_PNF  Literal    230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_TYPE_SCRIPT_PROMPT	Literal	230	
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_TYPE_SCRIPT_SKIP	Literal	230	
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_SEQUENCE_ERROR	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_START_ERR  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DS$K_TYPE_START_LIST	Literal	230	
	Literal	252	
	Literal	265	
	Literal	280	
	Literal	287	
	Literal	300	
	Literal	307	
	Literal	323	
	Literal	330	
	Literal	343	
	Literal	358	
	Literal	376	
	Literal	389	
	Literal	404	

Symbol	Type	Defined	Referenced ...
	Literal		452
	Literal		474
	Literal		488
	Literal		501
	Literal		508
	Literal		522
	Literal		529
	Literal		542
	Literal		556
	Literal		578
	Literal		591
	Literal		606
	Literal		613
	Literal		626
	Literal		633
	Literal		647
	Literal		654
	Literal		667
	Literal		685
	Literal		707
	Literal		721
	Literal		734
	Literal		741
	Literal		755
	Literal		762
	Literal		775
	Literal		789
	Literal		811
	Literal		824
	Literal		839
	Literal		846
	Literal		859
	Literal		866
	Literal		880
	Literal		887
	Literal		900
	Literal		951
	Literal		1053
	Literal		1061
DS$K_TYPE_SUMMARY	Literal		230
	Literal		252
	Literal		265
	Literal		280
	Literal		287
	Literal		300
	Literal		307
	Literal		323
	Literal		330
	Literal		343
	Literal		358
	Literal		376
	Literal		389
	Literal		404

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     452
    Literal     474
    Literal     488
    Literal     501
    Literal     508
    Literal     522
    Literal     529
    Literal     542
    Literal     556
    Literal     578
    Literal     591
    Literal     606
    Literal     613
    Literal     626
    Literal     633
    Literal     647
    Literal     654
    Literal     667
    Literal     685
    Literal     707
    Literal     721
    Literal     734
    Literal     741
    Literal     755
    Literal     762
    Literal     775
    Literal     789
    Literal     811
    Literal     824
    Literal     839
    Literal     846
    Literal     859
    Literal     866
    Literal     880
    Literal     887
    Literal     900
    Literal     951
    Literal    1053
    Literal    1061
DS$K_TYPE_USER_PROMPT  Literal     230
    Literal     252
    Literal     265
    Literal     280
    Literal     287
    Literal     300
    Literal     307
    Literal     323
    Literal     330
    Literal     343
    Literal     358
    Literal     376
    Literal     389
    Literal     404
```

Symbol	Type	Defined	Referenced ...
	Literal	452	
	Literal	474	
	Literal	488	
	Literal	501	
	Literal	508	
	Literal	522	
	Literal	529	
	Literal	542	
	Literal	556	
	Literal	578	
	Literal	591	
	Literal	606	
	Literal	613	
	Literal	626	
	Literal	633	
	Literal	647	
	Literal	654	
	Literal	667	
	Literal	685	
	Literal	707	
	Literal	721	
	Literal	734	
	Literal	741	
	Literal	755	
	Literal	762	
	Literal	775	
	Literal	789	
	Literal	811	
	Literal	824	
	Literal	839	
	Literal	846	
	Literal	859	
	Literal	866	
	Literal	880	
	Literal	887	
	Literal	900	
	Literal	951	
	Literal	1053	
	Literal	1061	
DSA$AL_APTMAIL	Bind	119	119
DSA$GL_APTCOM	Bind	119	
DSA$GL_DEVLEN	Bind	119	
DSA$GL_DEVNAM	Bind	119	
DSA$GL_ERRNO	Bind	119	
DSA$GL_EVENT	Bind	119	
DSA$GL_FLAGS	Bind	119	
DSA$GL_MSGPTR	Bind	119	
DSA$GL_MSGTYP	Bind	119	
DSA$GL_PASSES	Bind	119	
DSA$GL_PASSNO	Bind	119	
DSA$GL_SECTNO	Bind	119	
DSA$GL_SID	Bind	119	
DSA$GL_SUBTNO	Bind	119	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DSA$GL_TESTNO   Bind       119
DSA$GL_UNITS   Bind       119
FALSE    Literal  Lib03  1000    1001    1002    1003    1004    1005
GET1ST_    Macro    Lib04   120     230     252     265     280     287     300     307     323     330
              343     358     376     389     404     452     474     488     501     508
              522     529     542     556     578     591     606     613     626     633
              647     654     667     685     707     721     734     741     755     762
              775     789     811     824     839     846     859     866     880     887
              900     951    1053    1061
GET2ND     Unbound          120     230     252     265     280     287     300     307     323     330
              343     358     376     389     404     452     474     488     501     508
              522     529     542     556     578     591     606     613     626     633
              647     654     667     685     707     721     734     741     755     762
              775     789     811     824     839     846     859     866     880     887
              900     951    1053    1061
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal     120
HS$K_ACTION_ADVANCE_GENERAL_COM Literal     120
HS$K_ACTION_ADVANCE_STATE Literal     120
HS$K_ACTION_CHANGE_TABLE Literal     120
HS$K_ACTION_DIAGNOSTIC_ERROR Literal     120
HS$K_ACTION_DONE_GENERAL_COMS Literal     120
HS$K_ACTION_DO_NOTHING  Literal     120
HS$K_ACTION_DUMMY_3  Literal     120
HS$K_ACTION_DUMMY_4  Literal     120
HS$K_ACTION_DUMMY_5  Literal     120
HS$K_ACTION_DUMMY_6  Literal     120
HS$K_ACTION_INIT_HS  Literal     120
HS$K_ACTION_INTERNAL_ERROR Literal     120
HS$K_ACTION_LAST_ACTION  Literal     120
HS$K_ACTION_LOAD_ERROR  Literal     120
HS$K_ACTION_LOAD_GENERAL_COM Literal     120
HS$K_ACTION_LOAD_LOAD_COM Literal     120
HS$K_ACTION_LOAD_SELECT_COM Literal     120
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     120
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     120
HS$K_ACTION_LOAD_START_COM Literal     120
HS$K_ACTION_PREPARATION_ERROR Literal     120
HS$K_ACTION_START_ERROR  Literal     120
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     120
HS$K_STATE_ADVANCE_GENERAL_COM Literal     120
HS$K_STATE_CHANGE_TABLE  Literal     120
HS$K_STATE_DIAGNOSTIC_ERROR Literal     120
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     120
HS$K_STATE_DONE_GENERAL_COMS Literal     120
HS$K_STATE_DUMMY_1  Literal     120
HS$K_STATE_DUMMY_2  Literal     120
HS$K_STATE_DUMMY_3  Literal     120
HS$K_STATE_DUMMY_4  Literal     120
HS$K_STATE_DUMMY_6  Literal     120
HS$K_STATE_INIT_HS  Literal     120
HS$K_STATE_INTERNAL_ERROR Literal     120
HS$K_STATE_LAST_STATE  Literal     120
HS$K_STATE_LOAD_ERROR  Literal     120
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_STATE_LOAD_GENERAL_COM Literal    120
HS$K_STATE_LOAD_LOAD_COM Literal    120
HS$K_STATE_LOAD_SELECT_COM Literal    120
HS$K_STATE_LOAD_SET_EVENT_COM Literal    120
HS$K_STATE_LOAD_SET_FLAGS_COM Literal    120
HS$K_STATE_LOAD_START_COM Literal    120
HS$K_STATE_PREPARATION_ERROR Literal    120
HS$K_STATE_START_ERROR  Literal    120
INSQUE    Builtin    215   267     289     309     332     391     490     510     531     593     615
             635     656     723     743     764     826     848     868     889
IRP$B_TYPE   Macro    Lib04   961
IRP$W_SIZE   Macro    Lib04   960
JSB_ALLOC   Linkage  Lib01  1038
JSB_DEALL   Linkage  Lib01   946
MEN$K_HS_STATE_TABLE  Literal    120
MEN$K_MM_STATE_TABLE  Literal    120
MEN$K_TQ_STATE_TABLE  Literal    120
MENU$K_EXIT_AUTOSIZER_ERROR Literal    120
MENU$K_EXIT_INTERNAL_ERROR Literal    120
MENU$K_EXIT_LAST_REASON  Literal    120
MENU$K_EXIT_OPERATOR_ABORT Literal    120
MENU$K_EXIT_OPERATOR_STOP Literal    120
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal    120
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal    120
MM$K_ACTION_ADVANCE_STATE Literal    120
MM$K_ACTION_AUTOSIZER_ERROR Literal    120
MM$K_ACTION_BUILD_DDBS  Literal    120
MM$K_ACTION_BUILD_HSTS  Literal    120
MM$K_ACTION_CHANGE_TABLE Literal    120
MM$K_ACTION_DONE_INITS  Literal    120
MM$K_ACTION_DO_NOTHING  Literal    120
MM$K_ACTION_DUMMY_3  Literal    120
MM$K_ACTION_DUMMY_4  Literal    120
MM$K_ACTION_DUMMY_5  Literal    120
MM$K_ACTION_DUMMY_6  Literal    120
MM$K_ACTION_DUMMY_7  Literal    120
MM$K_ACTION_DUMMY_8  Literal    120
MM$K_ACTION_INTERNAL_ERROR Literal    120
MM$K_ACTION_LAST_ACTION  Literal    120
MM$K_ACTION_LOAD_AUTOSIZER Literal    120
MM$K_ACTION_MENU_PROMPT  Literal    120
MM$K_ACTION_PRINT_MENU  Literal    120
MM$K_ACTION_PRINT_MENU_HEADING Literal    120
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal    120
MM$K_ACTION_START_AUTOSIZER Literal    120
MM$K_STATE_AUTOSIZER_ERROR Literal    120
MM$K_STATE_AUTOSIZER_RUNNING Literal    120
MM$K_STATE_BUILD_DDBS  Literal    120
MM$K_STATE_BUILD_HSTS  Literal    120
MM$K_STATE_CHANGE_TABLE  Literal    120
MM$K_STATE_DONE_INITS  Literal    120
MM$K_STATE_DUMMY_1  Literal    120
MM$K_STATE_DUMMY_2  Literal    120
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_STATE_DUMMY_3  Literal     120
MM$K_STATE_DUMMY_5  Literal     120
MM$K_STATE_DUMMY_7  Literal     120
MM$K_STATE_DUMMY_8  Literal     120
MM$K_STATE_INTERNAL_ERROR Literal     120
MM$K_STATE_LAST_STATE  Literal     120
MM$K_STATE_LOAD_AUTOSIZER Literal     120
MM$K_STATE_MENU_PROMPT  Literal     120
MM$K_STATE_PRINT_MENU  Literal     120
MM$K_STATE_PRINT_MENU_HEADING Literal     120
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     120
MM$K_STATE_START  Literal     120
MM$K_STATE_START_AUTOSIZER Literal     120
MODNAM    Macro    Lib01   136
NUL2ND_     Macro    Lib04   120     230     252     265     280     287     300     307     323     330
              343     358     376     389     404     452     474     488     501     508
              522     529     542     556     578     591     606     613     626     633
              647     654     667     685     707     721     734     741     755     762
              775     789     811     824     839     846     859     866     880     887
              900     951    1053    1061
PREV_DDB_PTR   Local       210   253=    267.    281=    289.    301=    309.    324=    332.    377=    391.
              405=    475=    490.    502=    510.    523=    531.    579=    593.    607=
              615.    627=    635.    648=    656.    708=    723.    735=    743.    756=
              764.    812=    826.    840=    848.    860=    868.    881=    889.
PRINT_DDB_STATUS  Macro       183   252     280     300     323     343     376     404     474     501     522
              542     578     606     626     647     667     707     734     755     775
              811     839     859     880     900
RETURNED_SIZE   Local      1035  1047=   1048.   1053.   1055.   1061.
SIZE    Unbound           198     199
    Local       209   230=    230a    249.    265=    265a    278.    287=    287a    297.    307=
              307a    320.    330=    330a    340.    358=    358a    373.    389=    389a
              402.    452=    452a    471.    488=    488a    498.    508=    508a    519.
              529=    529a    539.    556=    556a    575.    591=    591a    604.    613=
              613a    623.    633=    633a    644.    654=    654a    664.    685=    685a
              704.    721=    721a    731.    741=    741a    752.    762=    762a    772.
              789=    789a    808.    824=    824a    837.    846=    846a    856.    866=
              866a    877.    887=    887a    897.
    RoutForm    913   951.    960.
    RoutForm   1009  1047a.  1048a.  1053a.  1055a=  1061a.
TQ$K_ACTION_ADVANCE_STATE Literal     120
TQ$K_ACTION_CHANGE_TABLE Literal     120
TQ$K_ACTION_DONE_TEST_QUEUE Literal     120
TQ$K_ACTION_DO_NOTHING  Literal     120
TQ$K_ACTION_DUMMY_3  Literal     120
TQ$K_ACTION_DUMMY_4  Literal     120
TQ$K_ACTION_DUMMY_5  Literal     120
TQ$K_ACTION_GET_DDB  Literal     120
TQ$K_ACTION_INIT_TQ  Literal     120
TQ$K_ACTION_INTERNAL_ERROR Literal     120
TQ$K_ACTION_LAST_ACTION  Literal     120
TQ$K_STATE_CHANGE_TABLE  Literal     120
TQ$K_STATE_DONE_TEST_QUEUE Literal     120
TQ$K_STATE_DUMMY_1  Literal     120
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
TQ$K_STATE_DUMMY_2  Literal     120
TQ$K_STATE_DUMMY_3  Literal     120
TQ$K_STATE_DUMMY_4  Literal     120
TQ$K_STATE_DUMMY_5  Literal     120
TQ$K_STATE_GET_DDB  Literal     120
TQ$K_STATE_INIT_TQ  Literal     120
TQ$K_STATE_INTERNAL_ERROR Literal     120
TQ$K_STATE_LAST_STATE  Literal     120
TRUE    Literal  Lib03   241     244     275     317     369     399     463     466     567     570
              601     696     699     800     803     834     998



  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDDDB.B32;1
    2  Module CRDDDB
   93  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   96  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   99  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
  102  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
 1066  Eludom CRDDDB


 KEY TO REFERENCE TYPE FLAGS

      . Fetch
      = Store
      c Routine call
      a Address use
      @ Indirect use
      e External, external routine, or external literal declaration
      f Forward or forward routine declaration
      h Condition handler enabling
      m Map declaration
      u Undeclare declaration


;     Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          4          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923          3          0          94          00:00.1
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486         35          7          62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          5          0        1000          00:04.3
```

```
;     COMMAND QUALIFIERS

; BLISS CRDDDB.B32/LIST=CRDDDB.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDDDB.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  3443 code + 2727 data bytes
; Run Time:    02:19.5
; Elapsed Time:    14:13.7
; Lines/CPU Min:     458
; Lexemes/CPU-Min: 71590
; Memory Used: 1063 pages
; Compilation Complete
```

```
;    0001   0 %TITLE '*** CRDGENCOM Module'
;    0002   0 MODULE CRDGENCOM (    ! Manipulate the DS general command table
;    0003   0    IDENT = '01-02'
;    0004   0    ) =
;    0005   0 !++
;    0006   0 ! COPYRIGHT (c) 1980, 1981
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the CRD-Loadable image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! Manipulate the DS general command table.
;    0032  0 !
;    0033  0 ! Author:
;    0034  0 !
;    0035  0 ! Jack Stansbury
;    0036  0 !
;    0037  0 ! Creation Date:
;    0038  0 !
;    0039  0 ! 14-April-1982
;    0040  0 !
;    0041  0 ! Version:
;    0042  0 !
;    0043  0 ! 6.7
;    0044  0 !
;    0045  0 ! Modified By:
;    0046  0 !
;    0047  0 ! 01 Jack Stansbury, Version 6.9, August 18, 1982
;    0048  0 !  Changed the  CLEAR FLAGS ALL  general command to be  SET FLAGS
;    0049  0 !  DEFAULT . This may clear up a problem with the Menu Test prompts.
;    0050  0 !
;    0051  0 ! 02 Jack Stansbury, Version 6.9, August 18, 1982
;    0052  0 !  Nope, the above didn't work because the BINARY flag is cleared by
;    0053  0 !  a SET FLAGS DEFAULT command.
;    0054  0 !--
```

```
;    0055  0 %SBTTL 'Libraries'
;    0056  1 BEGIN      ! Begin the CRDGENCOM module
;    0057  1
;    0058  1 LIBRARY
;    0059  1   '$DS';     ! Use Ds.L32 first
;    0060  1
;    0061  1 LIBRARY
;    0062  1   '$Diag';   ! Use Diag.L32 second
;    0063  1
;    0064  1 LIBRARY
;    0065  1   '$CRD';    ! Use CRD.L32 third
;    0066  1
;    0067  1 LIBRARY
;    0068  1   'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0069  1 %SBTTL 'Table of Contents'
;    0070  1
;    0071  1 FORWARD ROUTINE
;    0072  1  CRD$Init_General_Com_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0073  1       ! Initialize the CRD$GA_General_Commands table
;    0074  1
;    0075  1  CRD$Get_General_Command     : ADDRESSING_MODE (LONG_RELATIVE),
;    0076  1       ! Get the CRD$GA_General_Commands command pointer
;    0077  1
;    0078  1  CRD$Print_Command_Table     : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
;    0079  1       ! Print the contents of the general command table
```

```
ZZ-ENSAB-2.0    Included Macros and Literals                    19-Sep-1985      Fiche 3  Frame E13        Sequence 572
CRDGENCOM *** CRDGENCOM Module    19-Sep-1985 16:49:22 VAX-11 Bliss-32 V4.1-746     Page   5
01-02  Included Macros and Literals    3-Jan-1985 17:02:04 [DS.CRD.V020]CRDGENCOM.B32;1   (5)

;    0080  1 %SBTTL 'Included Macros and Literals'
;    0081  1
;    0082  1  $CRD_Literals;    ! Define the CRD literals
```

```
;    0083  1 %SBTTL 'Psect Definitions'
;    0084  1
;    0085  1 PSECT
;    0086  1     PLIT   = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0087  1
;    0088  1 PSECT
;    0089  1     CODE   = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0090  1
;    0091  1 PSECT
;    0092  1     OWN    = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0093  1
;    0094  1 PSECT
;    0095  1     GLOBAL = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0096  1 %SBTTL 'Module-Global Static Storage'
;    0097  1
;    0098  1  ModNam ('CRDGENCOM');  ! Module name for ErrSup macro
```

```
;    0099  1 %SBTTL 'Global Own Storage'
;    0100  1
;    0101  1 GLOBAL
;    0102  1  CRD$GL_Current_General_Com; ! The current general command pointer
```

```
;   0103  1 %SBTTL 'Local Macros'
;   0104  1
;   0105  1 MACRO
; M 0106  1  DS_Coms (DS_Command_String) =
; M 0107  1   %EXACTSTRING (CRD$K_DS_Command_Size, %C' ', DS_Command_String)
;   0108  1  %;
```

```
;    0109  1 %SBTTL 'Module-Global Bindings'
;    0110  1
;    0111  1 GLOBAL BIND
;    0112  1  CRD$GA_General_Commands =
;    0113  1   UPLIT (
;    0114  1    DS_Coms ('CLEAR FLAGS ALL'),          ![02]
;    0115  1    DS_Coms ('CLEAR EVENT FLAGS ALL'),
;    0116  1    DS_Coms ('DESELECT ALL')
;    0117  1         )
;    0118  1   : General_Command_Structure [CRD$K_Numb_General_Commands];
;    0119  1      ! This table contains the General DS Commands
```

```
;    0120  1 %SBTTL 'CRD$Init_General_Com_Table'
;    0121  1
;    0122  1 GLOBAL ROUTINE CRD$Init_General_Com_Table : NOVALUE =
;    0123  1
;    0124  1 !++
;    0125  1 ! Functional Description:
;    0126  1 !
;    0127  1 ! Initialize the data structures necessary to manipulate the CRD$GA_General_Commands table.
;    0128  1 !
;    0129  1 ! Formal Input Parameters:
;    0130  1 !
;    0131  1 ! None
;    0132  1 !
;    0133  1 ! Formal Output Parameters:
;    0134  1 !
;    0135  1 ! None
;    0136  1 !
;    0137  1 ! Side Effects:
;    0138  1 !
;    0139  1 ! None
;    0140  1 !
;    0141  1 ! Completion Codes:
;    0142  1 !
;    0143  1 ! None
;    0144  1 !--
```

```
;   0145  2 BEGIN     ! Routine CRD$Init_General_Com_Table
;   0146  2
;   0147  2  CRD$GL_Current_General_Com = 0; ! Initialize to point to first one
;   0148  2
;   0149  1 END;      ! Routine CRD$Init_General_Com_Table


            .TITLE  CRDGENCOM *** CRDGENCOM Module
            .IDENT  \01-02\

            .PSECT  WORK,NOEXE,2

            00000 CRD$GL_CURRENT_GENERAL_COM::
            .BLKB   4

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      4D  4F  43  4E  45  47  44  52  43  09  00000 P.AAA:  .ASCII  <9>\CRDGENCOM\          ;
            0000A    .BLKB   2
4C  4C  41  20  53  47  41  4C  46  20  52  41  45  4C  43  0000C P.AAB:  .ASCII  \CLEAR FLAGS ALL                    \  ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  0001B              ;
      20  20  20  20  20  20  20  20  20  20  0002A              ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  00034    .ASCII  \                                    \  ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  00043              ;
      20  20  20  20  20  20  20  20  20  20  00052              ;
41  4C  46  20  54  4E  45  56  45  20  52  41  45  4C  43  0005C    .ASCII  \CLEAR EVENT FLAGS ALL                \  ;
20  20  20  20  20  20  20  20  20  4C  4C  41  20  53  47  0006B              ;
      20  20  20  20  20  20  20  20  20  20  0007A              ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  00084    .ASCII  \                                    \  ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  00093              ;
      20  20  20  20  20  20  20  20  20  20  000A2              ;
20  20  20  4C  4C  41  20  54  43  45  4C  45  53  45  44  000AC    .ASCII  \DESELECT ALL                        \  ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  000BB              ;
      20  20  20  20  20  20  20  20  20  20  000CA              ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  000D4    .ASCII  \                                    \  ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  000E3              ;
      20  20  20  20  20  20  20  20  20  20  000F2              ;

          $MODULE=       P.AAA
          CRD$GA_GENERAL_COMMANDS==
                P.AAB
           .EXTRN  CRD$INIT_GENERAL_COM_TABLE
           .EXTRN  CRD$GET_GENERAL_COMMAND
           .EXTRN  CRD$PRINT_COMMAND_TABLE

           .PSECT  CODE,NOWRT,  SHR,  PIC,2

             0000 00000    .ENTRY  CRD$INIT_GENERAL_COM_TABLE, Save nothing    ; 0122
      00000000'  EF  D4 00002    CLRL    CRD$GL_CURRENT_GENERAL_COM        ; 0147
       04 00008    RET                 ; 0149

; Routine Size:  9 bytes,    Routine Base:  CODE + 0000
```

```
;    0150  1 %SBTTL 'CRD$Get_General_Command'
;    0151  1
;    0152  1 GLOBAL ROUTINE CRD$Get_General_Command =
;    0153  1
;    0154  1 !++
;    0155  1 ! Functional Description:
;    0156  1 !
;    0157  1 ! Get a pointer to the general command.
;    0158  1 !
;    0159  1 ! Formal Input Parameters:
;    0160  1 !
;    0161  1 ! None
;    0162  1 !
;    0163  1 ! Formal Output Parameters:
;    0164  1 !
;    0165  1 ! None
;    0166  1 !
;    0167  1 ! Side Effects:
;    0168  1 !
;    0169  1 ! None
;    0170  1 !
;    0171  1 ! Completion Codes:
;    0172  1 !
;    0173  1 ! None - A pointer is returned.
;    0174  1 !--
```

ZC0 S-MMMMMMMMMMNOOOTTT

```
;    0175  2 BEGIN     ! Routine CRD$Get_General_Command
;    0176  2
;    0177  3  RETURN (CRD$GA_General_Commands [.CRD$GL_Current_General_Com])
;    0178  3
;    0179  1 END;      ! Routine CRD$Get_General_Command




                0000 00000     .ENTRY  CRD$GET_GENERAL_COMMAND, Save nothing       ; 0152
               50 00000000'  EF 00000050   8F  C5 00002     MULL3   #80, CRD$GL_CURRENT_GENERAL_COM, R0       ; 0177
               03 00000000'  EF  D1 0000E     CMPL    CRD$GL_CURRENT_GENERAL_COM, #3       ;
               04  19 00015     BLSS    1$             ;
               50       01  CE 00017     MNEGL   #1, R0          ;
            04 0001A     RET                     ;
               50 00000000'EF40  9E 0001B 1$:   MOVAB   CRD$GA_GENERAL_COMMANDS[R0], R0       ;
            04 00023     RET                     ; 0179

; Routine Size:  36 bytes,    Routine Base:  CODE + 0009
```

```
;    0180  1 %SBTTL 'CRD$Print_Command_Table'
;    0181  1
;    0182  1 GLOBAL ROUTINE CRD$Print_Command_Table : NOVALUE =
;    0183  1
;    0184  1 !++
;    0185  1 ! Functional Description:
;    0186  1 !
;    0187  1 ! Print the contents of the general command table.
;    0188  1 !
;    0189  1 ! Formal Input Parameters:
;    0190  1 !
;    0191  1 ! None
;    0192  1 !
;    0193  1 ! Formal Output Parameters:
;    0194  1 !
;    0195  1 ! None
;    0196  1 !
;    0197  1 ! Side Effects:
;    0198  1 !
;    0199  1 ! None
;    0200  1 !
;    0201  1 ! Completion Codes:
;    0202  1 !
;    0203  1 ! None
;    0204  1 !
```

```
;    0205  2 BEGIN     ! Routine CRD$Print_Command_Tables
;    0206  2
;    0207  2  BIND
;    0208  2   Out_GC_Header = $ASCIC ('!/!/****************** The General Command Table ******************!/'),
;    0209  2   Out_GC_Table  = $ASCIC ('CRD$GA_General_Commands  [!1UL]:!/  - .!AD.!/'),
;    0210  2   Out_Footer    = $ASCIC ('!/!/');
;    0211  2
;    0212  2  $CRD_Print (Out_GC_Header);
;    0213  2
;    0214  2  INCR GC_Com FROM 0 TO (CRD$K_Numb_General_Commands - 1) DO
;    0215  3   BEGIN
; P  0216  3   $CRD_Print (
; P  0217  3    Out_GC_Table,
; P  0218  3    .GC_Com,
; P  0219  3    CRD$K_DS_Command_Size,
; P  0220  3    CRD$GA_General_Commands [.GC_Com]
;    0221  3    );
;    0222  2   END;
;    0223  2
;    0224  2  $CRD_Print (Out_Footer);
;    0225  1 END;     ! Routine CRD$Print_Command_Tables



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2F  21  2F  21  45  000FC P.AAC:  .ASCII  \E!/!/****************** The General Comm\  ;
65  47  20  65  68  54  20  2A  2A  2A  2A  2A  2A  2A  2A  0010B                                                          ;
      6D  6D  6F  43  20  6C  61  72  65  6E  0011A                                                                      ;
2A  2A  2A  2A  2A  20  65  6C  62  61  54  20  64  6E  61  00124    .ASCII  \and Table ******************!/\       ;
2F  21  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  00133                                                       ;
6C  61  72  65  6E  65  47  5F  41  47  24  44  52  43  2D  00142 P.AAD:  .ASCII  \-CRD$GA_General_Commands  [!1UL]:!/  - .\  ;
55  31  21  5B  20  20  73  64  6E  61  6D  6D  6F  43  5F  00151                                                       ;
    2E  20  2D  20  20  2F  21  3A  5D  4C  00160                                                                       ;
      2F  21  2E  44  41  21  0016A    .ASCII  \!AD.!/\          ;
    2F  21  2F  21  04  00170 P.AAE:  .ASCII  <4>\!/!/\         ;

        OUT_GC_HEADER=      P.AAC
        OUT_GC_TABLE=       P.AAD
        OUT_FOOTER=       P.AAE
         .EXTRN  CRD$PRINT

         .PSECT  CODE,NOWRT,  SHR,  PIC,2

           0004 00000     .ENTRY  CRD$PRINT_COMMAND_TABLE, Save R2       ; 0182
00000000'  EF  9F 00002     PUSHAB  OUT_GC_HEADER                         ; 0212
     01  DD 00008     PUSHL   #1                                        ;
     1A  DD 0000A     PUSHL   #26                                       ;
00000000G  FF       03  FB 0000C     CALLS   #3, @CRD$PRINT            ;
     52  D4 00013     CLRL    GC_COM                                    ; 0214
     50         52 00000050    8F  C5 00015 1$:   MULL3   #80, GC_COM, R0         ; 0221
     03       52  D1 0001D     CMPL    GC_COM, #3                       ;
     05  19 00020     BLSS    2$                                        ;
     7E       01  CE 00022     MNEGL   #1, -(SP)                        ;
     0A  11 00025     BRB     3$                                        ;
```

```
                50 00000000'EF40  9E 00027 2$:   MOVAB   CRD$GA_GENERAL_COMMANDS[R0], R0       ;
                50  DD 0002F     PUSHL   R0              ;
                7E 50   8F  9A 00031 3$:   MOVZBL  #80, -(SP)          ;
                52  DD 00035     PUSHL   GC_COM          ;
            00000000'  EF  9F 00037     PUSHAB  OUT_GC_TABLE          ;
                01  DD 0003D     PUSHL   #1              ;
                1A  DD 0003F     PUSHL   #26             ;
            00000000G  FF       06  FB 00041     CALLS   #6, aCRD$PRINT          ;
                C9        52        02  F3 00048     AOBLEQ  #2, GC_COM, 1$          ; 0214
            00000000'  EF  9F 0004C     PUSHAB  OUT_FOOTER          ; 0224
                01  DD 00052     PUSHL   #1              ;
                1A  DD 00054     PUSHL   #26             ;
            00000000G  FF       03  FB 00056     CALLS   #3, aCRD$PRINT          ;
             04 0005D     RET                 ; 0225

; Routine Size:  94 bytes,    Routine Base:  CODE + 002D
```

```
;   0226  1 END     ! End of CRDGENCOM module
;   0227  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA           373  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK             4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE           139  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
Symbol                            Type  Defined  Referenced ...
--------------------------------- -------- ----- ---------------
$ASCIC    Macro    Lib02   208     209     210
$CRD_LITERALS   Macro    Lib03    82
$CRD_PRINT   Macro    Lib03   212     216     224
$DS_TYPEDEF   Macro    Lib02   212     221     224
$EQULST    Macro    Lib04    82     212     221     224
$ER    Comptime    98
$MODULE    Bind        98
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     82
AUTO$K_EXIT_CONTROL_C  Literal     82
AUTO$K_EXIT_DUMMY_2  Literal     82
AUTO$K_EXIT_DUMMY_3  Literal     82
AUTO$K_EXIT_ERRORS_COMPLETE Literal     82
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     82
AUTO$K_EXIT_INTERNAL_ERROR Literal     82
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     82
AUTO$K_EXIT_LAST_REASON  Literal     82    82
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     82
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     82
AUTO$K_EXIT_NOERRORS_PREP Literal     82
CRD$GA_GENERAL_COMMANDS  External Lib03
    GlobBind   112   177     221     221a
CRD$GET_GENERAL_COMMAND  GlobRout   152 Lib03e     75f
CRD$GL_CURRENT_GENERAL_COM Global     102 Lib03e    147=    177.
CRD$INIT_GENERAL_COM_TABLE GlobRout   122 Lib03e     72f
CRD$K_ACTION_EQL_DO_NOTHING Literal     82
CRD$K_ACTION_RANGE_ERROR Literal     82
CRD$K_ADVANCE_DIAGNOSTIC Literal     82
CRD$K_ADVANCE_GENERAL_COM Literal     82
CRD$K_ADVANCE_STATE  Literal     82
CRD$K_ALLOCATION_ERROR  Literal     82
CRD$K_ASSIGN_PTABLE_PTRS Literal     82
CRD$K_AUTOSIZER_ERROR  Literal     82
CRD$K_AUTOSIZER_SET_FLAGS Literal     82
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_BAD_ACTION_NUMBER  Literal     82
CRD$K_BAD_TYPECODE_ERROR Literal     82
CRD$K_BASE_SYSTEM_IDC  Literal     82
CRD$K_BASE_SYSTEM_LESI  Literal     82
CRD$K_BASE_SYSTEM_NULL  Literal     82
CRD$K_BASE_SYSTEM_UDA_0  Literal     82
CRD$K_BASE_SYSTEM_UDA_1  Literal     82
CRD$K_BASE_SYSTEM_UDA_2  Literal     82
CRD$K_BASE_SYSTEM_UDA_3  Literal     82
CRD$K_CRD_INITIALIZATION Literal     82
CRD$K_CREATE_HST  Literal     82
CRD$K_DATA_LENGTH_ERROR  Literal     82
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     82
CRD$K_DDB_BLINK   Literal     82
CRD$K_DDB_FLINK   Literal     82
CRD$K_DDB_LAST_ENTRY  Literal     82
CRD$K_DDB_MEMORY_SIZE  Literal     82
CRD$K_DDB_MENU_DEVICE_NUMB Literal     82
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     82
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     82
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     82
CRD$K_DDB_PTABLE_ENTRY  Literal     82
CRD$K_DDB_STATUS  Literal     82
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     82
CRD$K_DEVICE_NAME  Literal     82
CRD$K_DIAGNOSTIC_ERROR  Literal     82
CRD$K_DIAGNOSTIC_NAME  Literal     82
CRD$K_DONE_GENERAL_COMS  Literal     82
CRD$K_DO_NOTHING  Literal     82
CRD$K_DS_COMMAND_SIZE  Unbound          107
    Literal  Lib03   114     115     116     221
CRD$K_DUMMY_10   Literal     82
CRD$K_DUMMY_11   Literal     82
CRD$K_DUMMY_12   Literal     82
CRD$K_DUMMY_13   Literal     82
CRD$K_DUMMY_3   Literal     82
CRD$K_DUMMY_4   Literal     82
CRD$K_DUMMY_5   Literal     82
CRD$K_DUMMY_6   Literal     82
CRD$K_DUMMY_7   Literal     82
CRD$K_DUMMY_8   Literal     82
CRD$K_DUMMY_9   Literal     82
CRD$K_EXIT_CRD   Literal     82
CRD$K_HOOK_POINT  Literal     82
CRD$K_HS_INTERNAL_ERROR  Literal     82
CRD$K_IDC_EVDLB   Literal     82
CRD$K_IDC_EVDLC   Literal     82
CRD$K_IDC_EVDLD   Literal     82
CRD$K_IDC_EVRAD_0  Literal     82
CRD$K_IDC_EVRAD_1  Literal     82
CRD$K_IDC_LAST_DIAGNOSTIC Literal     82
CRD$K_INTERNAL_ERROR  Literal     82
CRD$K_LAST_ACTION_ROUTINE Literal     82
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_LAST_ENTRY  Literal     82
CRD$K_LAST_FUNCTION  Literal     82
CRD$K_LAST_HOOK_POINT  Literal     82
CRD$K_LAST_INTERNAL_ERROR Literal     82
CRD$K_LAST_TYPE   Literal     82
CRD$K_LESI_ENWCA  Literal     82
CRD$K_LESI_EVRMB_0  Literal     82
CRD$K_LESI_EVRMB_1  Literal     82
CRD$K_LESI_LAST_DIAGNOSTIC Literal     82
CRD$K_LEVEL_4_PASSED  Literal     82
CRD$K_LOAD_AUTOSIZER  Literal     82
CRD$K_LOAD_ERROR  Literal     82
CRD$K_LOAD_GENERAL_COM  Literal     82
CRD$K_LOAD_LOAD_COM  Literal     82
CRD$K_LOAD_SELECT_COM  Literal     82
CRD$K_LOAD_SET_EVENT_COM Literal     82
CRD$K_LOAD_SET_FLAGS_COM Literal     82
CRD$K_LOAD_START_COM  Literal     82
CRD$K_LOAD_SUP_FILE  Literal     82
CRD$K_MM_INTERNAL_ERROR  Literal     82
CRD$K_NEW_LINE   Literal     82
CRD$K_NUMB_GENERAL_COMMANDS Literal  Lib03   118     214
CRD$K_PREPARATION_ERROR  Literal     82
CRD$K_PREPARATION_ERROR_TEXT Literal     82
CRD$K_RUNTIME   Literal     82
CRD$K_SAME_LINE   Literal     82
CRD$K_SET_EVENT_ARGS  Literal     82
CRD$K_SET_FLAGS_ARGS  Literal     82
CRD$K_SET_UP_DIAGNOSTIC  Literal     82
CRD$K_START   Literal     82
CRD$K_START_AUTOSIZER  Literal     82
CRD$K_START_ERROR  Literal     82
CRD$K_START_QUALIFIERS  Literal     82
CRD$K_STAR_LINE   Literal     82
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal     82
CRD$K_STATE_ADVANCE_GENERAL_COM Literal     82
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal     82
CRD$K_STATE_AUTOSIZER_ERROR Literal     82
CRD$K_STATE_AUTOSIZER_RUNNING Literal     82
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal     82
CRD$K_STATE_DIAGNOSTIC_ERROR Literal     82
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal     82
CRD$K_STATE_DONE_DIAGNOSTICS Literal     82
CRD$K_STATE_DONE_GENERAL_COMS Literal     82
CRD$K_STATE_DUMMY_1  Literal     82
CRD$K_STATE_DUMMY_10  Literal     82
CRD$K_STATE_DUMMY_12  Literal     82
CRD$K_STATE_DUMMY_13  Literal     82
CRD$K_STATE_DUMMY_2  Literal     82
CRD$K_STATE_DUMMY_3  Literal     82
CRD$K_STATE_DUMMY_4  Literal     82
CRD$K_STATE_DUMMY_6  Literal     82
CRD$K_STATE_DUMMY_7  Literal     82
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_DUMMY_8  Literal     82
CRD$K_STATE_EXIT_CRD  Literal     82
CRD$K_STATE_INTERNAL_ERROR Literal     82
CRD$K_STATE_LAST_STATE  Literal     82
CRD$K_STATE_LEVEL_4_PASSED Literal     82
CRD$K_STATE_LOAD_AUTOSIZER Literal     82
CRD$K_STATE_LOAD_ERROR  Literal     82
CRD$K_STATE_LOAD_GENERAL_COM Literal     82
CRD$K_STATE_LOAD_LOAD_COM Literal     82
CRD$K_STATE_LOAD_SELECT_COM Literal     82
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     82
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     82
CRD$K_STATE_LOAD_START_COM Literal     82
CRD$K_STATE_PREPARATION_ERROR Literal     82
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     82
CRD$K_STATE_START  Literal     82
CRD$K_STATE_START_AUTOSIZER Literal     82
CRD$K_STATE_START_ERROR  Literal     82
CRD$K_TEST_START_TEXT  Literal     82
CRD$K_TQ_INTERNAL_ERROR  Literal     82
CRD$K_TYPECODE   Literal     82
CRD$K_UDA_0_EVDLB  Literal     82
CRD$K_UDA_0_EVDLC  Literal     82
CRD$K_UDA_0_EVDLD  Literal     82
CRD$K_UDA_0_EVRLA_0  Literal     82
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     82
CRD$K_UDA_1_EVDLB  Literal     82
CRD$K_UDA_1_EVDLC  Literal     82
CRD$K_UDA_1_EVDLD  Literal     82
CRD$K_UDA_1_EVRLA_0  Literal     82
CRD$K_UDA_1_EVRLA_1  Literal     82
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     82
CRD$K_UDA_2_EVDLB  Literal     82
CRD$K_UDA_2_EVDLC  Literal     82
CRD$K_UDA_2_EVDLD  Literal     82
CRD$K_UDA_2_EVRLA_0  Literal     82
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     82
CRD$K_UDA_3_EVDLB  Literal     82
CRD$K_UDA_3_EVDLC  Literal     82
CRD$K_UDA_3_EVDLD  Literal     82
CRD$K_UDA_3_EVRLA_0  Literal     82
CRD$K_UDA_3_EVRLA_1  Literal     82
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     82
CRD$PRINT    External Lib03   212ac   221ac   224ac
CRD$PRINT_COMMAND_TABLE  GlobRout   182 Lib03e     78f
DS$K_PRINTB   Literal    212
    Literal     221
    Literal     224
DS$K_PRINTF   Literal    212   212
    Literal     221   221
    Literal     224   224
DS$K_PRINTI   Literal    212
    Literal     221
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- -------   ------
    Literal    224
DS$K_PRINTX   Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ABORT_PROGRAM  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ABORT_TEST  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_COMMAND_ERR  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_COMMAND_OUT  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_CRD_AUTOTEST  Literal    212   212
    Literal    221   221
    Literal    224   224
DS$K_TYPE_DS_PROMPT  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_DS_START  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ERRDEV  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ERRHARD  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ERROR_BODY  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ERROR_END  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ERRPREP  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ERRSOFT  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ERRSUP  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ERRSYS  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_ERR_HALT  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_EXCEPTION  Literal    212
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     221
    Literal     224
DS$K_TYPE_EXCEPTION_HEAD Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_FIRST_PASS  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_GENERAL  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_GENERAL_ERROR  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_NO_TESTS  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_PARAM_ERROR  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_PROGRAM_END  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_PROGRAM_INFO  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_PROGRAM_START  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_QIO_INVADP  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_QIO_NODRIVER  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_QIO_WRONGVER  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_SCRIPT_ECHO  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_SCRIPT_PNF  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_SCRIPT_PROMPT  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_SCRIPT_SKIP  Literal     212
    Literal     221
    Literal     224
DS$K_TYPE_SEQUENCE_ERROR Literal     212
    Literal     221
    Literal     224
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DS$K_TYPE_START_ERR  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_START_LIST  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_SUMMARY  Literal    212
    Literal    221
    Literal    224
DS$K_TYPE_USER_PROMPT  Literal    212
    Literal    221
    Literal    224
DS_COMMAND_STRING  MacrForm   106   107
DS_COMS    Macro      106   114     115     116
GC_COM    Local      214   221.
GENERAL_COMMAND_STRUCTURE Structur Lib03   118
GET1ST_    Macro    Lib04    82     212     221     224
GET2ND_    Unbound           82     212     221     224
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal     82
HS$K_ACTION_ADVANCE_GENERAL_COM Literal     82
HS$K_ACTION_ADVANCE_STATE Literal     82
HS$K_ACTION_CHANGE_TABLE Literal     82
HS$K_ACTION_DIAGNOSTIC_ERROR Literal     82
HS$K_ACTION_DONE_GENERAL_COMS Literal     82
HS$K_ACTION_DO_NOTHING  Literal     82
HS$K_ACTION_DUMMY_3  Literal     82
HS$K_ACTION_DUMMY_4  Literal     82
HS$K_ACTION_DUMMY_5  Literal     82
HS$K_ACTION_DUMMY_6  Literal     82
HS$K_ACTION_INIT_HS  Literal     82
HS$K_ACTION_INTERNAL_ERROR Literal     82
HS$K_ACTION_LAST_ACTION  Literal     82
HS$K_ACTION_LOAD_ERROR  Literal     82
HS$K_ACTION_LOAD_GENERAL_COM Literal     82
HS$K_ACTION_LOAD_LOAD_COM Literal     82
HS$K_ACTION_LOAD_SELECT_COM Literal     82
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     82
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     82
HS$K_ACTION_LOAD_START_COM Literal     82
HS$K_ACTION_PREPARATION_ERROR Literal     82
HS$K_ACTION_START_ERROR  Literal     82
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     82
HS$K_STATE_ADVANCE_GENERAL_COM Literal     82
HS$K_STATE_CHANGE_TABLE  Literal     82
HS$K_STATE_DIAGNOSTIC_ERROR Literal     82
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     82
HS$K_STATE_DONE_GENERAL_COMS Literal     82
HS$K_STATE_DUMMY_1  Literal     82
HS$K_STATE_DUMMY_2  Literal     82
HS$K_STATE_DUMMY_3  Literal     82
HS$K_STATE_DUMMY_4  Literal     82
HS$K_STATE_DUMMY_6  Literal     82
HS$K_STATE_INIT_HS  Literal     82
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------------- ------- ----- ---------------
HS$K_STATE_INTERNAL_ERROR Literal      82
HS$K_STATE_LAST_STATE  Literal      82
HS$K_STATE_LOAD_ERROR  Literal      82
HS$K_STATE_LOAD_GENERAL_COM Literal      82
HS$K_STATE_LOAD_LOAD_COM Literal      82
HS$K_STATE_LOAD_SELECT_COM Literal      82
HS$K_STATE_LOAD_SET_EVENT_COM Literal      82
HS$K_STATE_LOAD_SET_FLAGS_COM Literal      82
HS$K_STATE_LOAD_START_COM Literal      82
HS$K_STATE_PREPARATION_ERROR Literal      82
HS$K_STATE_START_ERROR  Literal      82
MEN$K_HS_STATE_TABLE  Literal      82
MEN$K_MM_STATE_TABLE  Literal      82
MEN$K_TQ_STATE_TABLE  Literal      82
MENU$K_EXIT_AUTOSIZER_ERROR Literal      82
MENU$K_EXIT_INTERNAL_ERROR Literal      82
MENU$K_EXIT_LAST_REASON  Literal      82
MENU$K_EXIT_OPERATOR_ABORT Literal      82
MENU$K_EXIT_OPERATOR_STOP Literal      82
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal      82
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal      82
MM$K_ACTION_ADVANCE_STATE Literal      82
MM$K_ACTION_AUTOSIZER_ERROR Literal      82
MM$K_ACTION_BUILD_DDBS  Literal      82
MM$K_ACTION_BUILD_HSTS  Literal      82
MM$K_ACTION_CHANGE_TABLE Literal      82
MM$K_ACTION_DONE_INITS  Literal      82
MM$K_ACTION_DO_NOTHING  Literal      82
MM$K_ACTION_DUMMY_3  Literal      82
MM$K_ACTION_DUMMY_4  Literal      82
MM$K_ACTION_DUMMY_5  Literal      82
MM$K_ACTION_DUMMY_6  Literal      82
MM$K_ACTION_DUMMY_7  Literal      82
MM$K_ACTION_DUMMY_8  Literal      82
MM$K_ACTION_INTERNAL_ERROR Literal      82
MM$K_ACTION_LAST_ACTION  Literal      82
MM$K_ACTION_LOAD_AUTOSIZER Literal      82
MM$K_ACTION_MENU_PROMPT  Literal      82
MM$K_ACTION_PRINT_MENU  Literal      82
MM$K_ACTION_PRINT_MENU_HEADING Literal      82
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal      82
MM$K_ACTION_START_AUTOSIZER Li eral      82
MM$K_STATE_AUTOSIZER_ERROR l   ral      82
MM$K_STATE_AUTOSIZER_RUNNING   teral      82
MM$K_STATE_BUILD_DDBS  Literal      82
MM$K_STATE_BUILD_HSTS  Literal      82
MM$K_STATE_CHANGE_TABLE  Literal      82
MM$K_STATE_DONE_INITS  Literal      82
MM$K_STATE_DUMMY_1  Literal      82
MM$K_STATE_DUMMY_2  Literal      82
MM$K_STATE_DUMMY_3  Literal      82
MM$K_STATE_DUMMY_5  Literal      82
MM$K_STATE_DUMMY_7  Literal      82
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_STATE_DUMMY_8  Literal     82
MM$K_STATE_INTERNAL_ERROR Literal     82
MM$K_STATE_LAST_STATE  Literal     82
MM$K_STATE_LOAD_AUTOSIZER Literal     82
MM$K_STATE_MENU_PROMPT  Literal     82
MM$K_STATE_PRINT_MENU  Literal     82
MM$K_STATE_PRINT_MENU_HEADING Literal     82
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     82
MM$K_STATE_START  Literal     82
MM$K_STATE_START_AUTOSIZER Literal     82
MODNAM    Macro    Lib01    98
NUL2ND    Macro    Lib04    82     212     221     224
OUT_FOOTER   Bind        210   224a
OUT_GC_HEADER   Bind        208   212a
OUT_GC_TABLE   Bind        209   221a
TQ$K_ACTION_ADVANCE_STATE Literal     82
TQ$K_ACTION_CHANGE_TABLE Literal     82
TQ$K_ACTION_DONE_TEST_QUEUE Literal     82
TQ$K_ACTION_DO_NOTHING  Literal     82
TQ$K_ACTION_DUMMY_3  Literal     82
TQ$K_ACTION_DUMMY_4  Literal     82
TQ$K_ACTION_DUMMY_5  Literal     82
TQ$K_ACTION_GET_DDB  Literal     82
TQ$K_ACTION_INIT_TQ  Literal     82
TQ$K_ACTION_INTERNAL_ERROR Literal     82
TQ$K_ACTION_LAST_ACTION  Literal     82
TQ$K_STATE_CHANGE_TABLE  Literal     82
TQ$K_STATE_DONE_TEST_QUEUE Literal     82
TQ$K_STATE_DUMMY_1  Literal     82
TQ$K_STATE_DUMMY_2  Literal     82
TQ$K_STATE_DUMMY_3  Literal     82
TQ$K_STATE_DUMMY_4  Literal     82
TQ$K_STATE_DUMMY_5  Literal     82
TQ$K_STATE_GET_DDB  Literal     82
TQ$K_STATE_INIT_TQ  Literal     82
TQ$K_STATE_INTERNAL_ERROR Literal     82
TQ$K_STATE_LAST_STATE  Literal     82


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDGENCOM.B32;1
    2  Module CRDGENCOM
   59  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   62  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   65  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   68  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  227  Eludom CRDGENCOM
```

```
 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     @ Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923          2          0          94          00:00.2
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486         11          2          62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0      1000          00:04.2




;     COMMAND QUALIFIERS

; BLISS CRDGENCOM.B32/LIST=CRDGENCOM.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDGENCOM.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  139 code + 377 data bytes
; Run Time:    00:23.1
; Elapsed Time:    00:43.9
; Lines/CPU Min:     590
; Lexemes/CPU-Min: 65479
; Memory Used: 163 pages
; Compilation Complete
```

```
;    0001  0
;    0002  0 %TITLE '*** CRDHDWSUP Module'
;    0003  0 MODULE CRDHDWSUP (    ! Contains the hardware support table and access routines
;    0004  0   IDENT = '01-05'
;    0005  0   ) =
;    0006  0 !++
;    0007  0 ! COPYRIGHT (c) 1980, 1981
;    0008  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0009  0 !
;    0010  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0011  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0012  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0013  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0014  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0015  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0016  0 ! REMAIN IN DEC.
;    0017  0 !
;    0018  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0019  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0020  0 ! CORPORATION.
;    0021  0 !
;    0022  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0023  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0024  0 !
```

```
;    0025  0 !++
;    0026  0 ! Facility:
;    0027  0 !
;    0028  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0029  0 !
;    0030  0 ! Abstract:
;    0031  0 !
;    0032  0 ! This module contains the Hardware_Support table as well as all associated
;    0033  0 ! data structures and accessing routines.
;    0034  0 !
;    0035  0 ! Author:
;    0036  0 !
;    0037  0 ! Jack Stansbury
;    0038  0 !
;    0039  0 ! Creation Date:
;    0040  0 !
;    0041  0 ! May 8, 1982
;    0042  0 !
;    0043  0 ! Version:
;    0044  0 !
;    0045  0 ! 6.9
;    0046  0 !
;    0047  0 ! Modified By:
;    0048  0 !
;    0049  0 ! 01 Jack Stansbury, Version 1.1, July 25, 1982
;    0050  0 !  Made some of the bindings local, instead of global.
;    0051  0 !
;    0052  0 ! 02 Jack Stansbury, Version 1.1, July 27, 1982
;    0053  0 !  Added the QUICK flag for EVDLC.
;    0054  0 !
;    0055  0 ! 03 Jack Stansbury, Version 1.1, August 20, 1982
;    0056  0 !  Took the EVSBA junk out of the Hardware Support Table.
;    0057  0 !
;    0058  0 ! 04 John Ciukaj, Version 1.2 6-Apr-1983
;    0059  0 !  - Added support for LESI/AZTEC Disk based
;    0060  0 !    and UDA/RA-disk based systems
;    0061  0 !
;    0062  0 ! 05 Scott Cohen Version 1.3 23-Sep-1983
;    0063  0 !  - Added ENWCA support for LCN PEARL system
;    0064  0 !  - Modified AZTEC generic name DAAn to DUAn ONLY FOR RELEASE 15
;    0065  0 !
;    0066  0 !--
```

```
;    0067  0 %SBTTL 'Libraries'
;    0068  1 BEGIN     ! Begin the CRDHDWSUP module
;    0069  1
;    0070  1 LIBRARY
;    0071  1   '$DS';     ! Use Ds.L32 first
;    0072  1
;    0073  1 LIBRARY
;    0074  1   '$Diag';    ! Use Diag.L32 second
;    0075  1
;    0076  1 LIBRARY
;    0077  1   '$CRD';     ! Use CRD.L32 third
;    0078  1
;    0079  1 LIBRARY
;    0080  1   'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0081  1 %SBTTL 'Table of Contents'
;    0082  1
;    0083  1 FORWARD ROUTINE
;    0084  1  CRD$Build_Support_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0085  1       ! Initialize the Hardware Support table
;    0086  1
;    0087  1  CRD$Print Hardware_Support : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
;    0088  1       ! Print the contents of the hardware support table
```

```
;    0089  1 %SBTTL 'Included Macros and Literals'
;    0090  1
;    0091  1  $CRD_Literals;    ! The CRD literals
```

```
;    0092  1 %SBTTL 'Psect Definitions'
;    0093  1
;    0094  1 PSECT
;    0095  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0096  1
;    0097  1 PSECT
;    0098  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0099  1
;    0100  1 PSECT
;    0101  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0102  1
;    0103  1 PSECT
;    0104  1     GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0105  1 %SBTTL 'Global Static Storage'
;    0106  1
;    0107  1  ModNam ('CRDHDWSUP');  ! Module name for ErrSup macro
```

```
;    0108  1 %SBTTL 'Global Own Storage'
;    0109  1
;    0110  1 GLOBAL
;    0111  1  CRD$GB_Base_System: BYTE INITIAL (BYTE (CRD$K_Base_System_Null)),
;    0112  1       ! Location contains code specifying
;    0113  1       ! CRD Base System type.  Initially,
;    0114  1       ! set to null.  After Autosizer is run
;    0115  1       ! the correct system type will be determined
;    0116  1       ! and denoted here.       [04]
;    0117  1  CRD$GA_Hdwr_Sprt_Table: LONG INITIAL (LONG (0)),
;    0118  1       ! Pointer to the Hardware Support Table
;    0119  1       ! to be used.  After  Autosizer is run
;    0120  1       ! the base system is determined, and
;    0121  1       ! on that basis the correct hardware
;    0122  1       ! support table pointer is loaded here.     [04]
;    0123  1  CRD$AL_IDC_Hdwr_Sprt_Table :
;    0124  1   Hardware_Support_Structure [CRD$K_IDC_Numb_Diagnostics, CRD$K_Numb_Support_Entries],
;    0125  1       ! Hardware Support Table for IDCbased system    [04]
;    0126  1  CRD$AL_LESI_Hdwr_Sprt_Table :
;    0127  1   Hardware_Support_Structure [CRD$K_LESI_Numb_Diagnostics, CRD$K_Numb_Support_Entries],
;    0128  1       ! Hardware Support Table for LESI/AZTEC based system   [04]
;    0129  1  CRD$AL_UDA_0_Hdwr_Sprt_Table :
;    0130  1   Hardware_Support_Structure [CRD$K_UDA_0_Numb_Diagnostics, CRD$K_Numb_Support_Entries],
;    0131  1       ! Hardware Support Table for UDA based system
;    0132  1       ! w th no RA60 Drive 1, RA80 or 81 Drive 0    [04]
;    0133  1  CRD$AL_UDA_1_Hdwr_Sprt_Table :
;    0134  1   Hardware_Support_Structure [CRD$K_UDA_1_Numb_Diagnostics, CRD$K_Numb_Support_Entries],
;    0135  1       ! Hardware Support Table for UDA based system
;    0136  1       ! with RA60 Drive 1, RA80 or 81 Drive 0     [04]
;    0137  1  CRD$AL_UDA_2_Hdwr_Sprt_Table :
;    0138  1   Hardware_Support_Structure [CRD$K_UDA_2_Numb_Diagnostics, CRD$K_Numb_Support_Entries],
;    0139  1       ! Hardware Support Table for UDA based system
;    0140  1       ! with no RA60 Drive 1, RA60 Drive 0     [04]
;    0141  1  CRD$AL_UDA_3_Hdwr_Sprt_Table :
;    0142  1   Hardware_Support_Structure [CRD$K_UDA_3_Numb_Diagnostics, CRD$K_Numb_Support_Entries];
;    0143  1       ! Hardware Support Table for UDA based system
;    0144  1       ! with RA60 Drive 1, RA60 Drive 0     [04]
```

```
;    0145  1 %SBTTL 'Local Bindings for IDC based system'
;    0146  1
;    0147  1 BIND                ![04]
;    0148  1
;    0149  1  !+
;    0150  1  ! This data for EVRAD is for the R80 or the RL02 (on DQA0) test.
;    0151  1  ! Leave the Test_Start_Text device name field blank as a default. The field will be f'lled
;    0152  1  ! in later if the device can be attached by the AutoSizer.
;    0153  1  !-
;    0154  1
;    0155  1  CRD$GT_IDC_EVRAD_0    = $ASCIC ('EVRAD'),
;    0156  1  CRD$GT_IDC_EVRAD_0_Set_Flgs = $ASCIC ('QUICK'),
;    0157  1  CRD$GT_IDC_EVRAD_0_Set_Evnt = $ASCIC (''),
;    0158  1  CRD$GT_IDC_EVRAD_0_Dev_Name = $ASCIC ('DQA0'),
;    0159  1 ! CRD$GT_IDC_EVRAD_0_Dev_Name      = $ASCIC ('TTA6'),
;    0160  1  CRD$GT_IDC_EVRAD_0_Strt_Qual = $ASCIC (''),
;    0161  1  CRD$GT_IDC_EVRAD_0_Tst_Strt  = $ASCIC ('              DQA0'),
;    0162  1  CRD$GT_IDC_EVRAD_0_RunTime  = $ASCIC ('0:30'),
;    0163  1  CRD$GT_IDC_EVRAD_0_Prep_Err = $ASCIC ('DQA0: DISK DRIVE TESTING INCOMPLETE'),
;    0164  1
;    0165  1  !+
;    0166  1  ! This data for EVRAD is for the RL02 (on DQA1) test.
;    0167  1  !-
;    0168  1
;    0169  1  CRD$GT_IDC_EVRAD_1    = $ASCIC ('EVRAD'),
;    0170  1  CRD$GT_IDC_EVRAD_1_Set_Flgs = $ASCIC ('QUICK'),
;    0171  1  CRD$GT_IDC_EVRAD_1_Set_Evnt = $ASCIC (''),
;    0172  1  CRD$GT_IDC_EVRAD_1_Dev_Name = $ASCIC ('DQA1'),
;    0173  1 ! CRD$GT_IDC_EVRAD_1_Dev_Name = $ASCIC ('TTA6'),
;    0174  1  CRD$GT_IDC_EVRAD_1_Strt_Qual = $ASCIC (''),
;    0175  1  CRD$GT_IDC_EVRAD_1_Tst_Strt = $ASCIC ('RL02 PART 2   DQA1'),
;    0176  1  CRD$GT_IDC_EVRAD_1_RunTime  = $ASCIC ('0:30'),
;    0177  1  CRD$GT_IDC_EVRAD_1_Prep_Err = $ASCIC ('DQA1: DISK DRIVE TESTING INCOMPLETE'),
;    0178  1
;    0179  1  !+
;    0180  1  ! This data for EVDLB is for the XGA0 (on the Combo board) test.
;    0181  1  !-
;    0182  1
;    0183  1  CRD$GT_IDC_EVDLB          = $ASCIC ('EVDLB'),
;    0184  1  CRD$GT_IDC_EVDLB_Set_Flgs    = $ASCIC (''),
;    0185  1  CRD$GT_IDC_EVDLB_Set_Evnt    = $ASCIC (''),
;    0186  1  CRD$GT_IDC_EVDLB_Dev_Name       = $ASCIC ('XGA0'),
;    0187  1 ! CRD$GT_IDC_EVDLB_Dev_Name       = $ASCIC ('TTA6'),
;    0188  1  CRD$GT_IDC_EVDLB_Strt_Qual  = $ASCIC (''),
;    0189  1  CRD$GT_IDC_EVDLB_Tst_Strt   = $ASCIC ('DMF32S        XGA0'),
;    0190  1  CRD$GT_IDC_EVDLB_RunTime       = $ASCIC ('1:00'),
;    0191  1  CRD$GT_IDC_EVDLB_Prep_Err   = $ASCIC ('XGA0: COMBO BOARD TESTING INCOMPLETE'),
;    0192  1
;    0193  1  !+
;    0194  1  ! This data for EVDLC is for the TXA (on the Combo board) test.
;    0195  1  !-
;    0196  1
;    0197  1  CRD$GT_IDC_EVDLC          = $ASCIC ('EVDLC'),
;    0198  1  CRD$GT_IDC_EVDLC_Set_Flgs    = $ASCIC ('QUICK'),
;    0199  1  CRD$GT_IDC_EVDLC_Set_Evnt    = $ASCIC (''),
```

```
;    0200  1  CRD$GT_IDC_EVDLC_Dev_Name       = $ASCIC ('TXA'),
;    0201  1 ! CRD$GT_IDC_EVDLC_Dev_Name       = $ASCIC ('TTA6'),
;    0202  1  CRD$GT_IDC_EVDLC_Strt_Qual  = $ASCIC (''),
;    0203  1  CRD$GT_IDC_EVDLC_Tst_Strt   = $ASCIC ('DMF32A   TXA0-TXA7'),
;    0204  1  CRD$GT_IDC_EVDLC_RunTime        = $ASCIC ('0:30'),
;    0205  1  CRD$GT_IDC_EVDLC_Prep Err   = $ASCIC ('TXA: COMBO BOARD TESTING INCOMPLETE'),
;    0206  1
;    0207  1  !+
;    0208  1  ! This data for EVDLD is for the LCA (on the Combo board) test.
;    0209  1  !-
;    0210  1
;    0211  1  CRD$GT_IDC_EVDLD          = $ASCIC ('EVDLD'),
;    0212  1  CRD$GT_IDC_EVDLD_Set_Flgs    = $ASCIC (''),
;    0213  1  CRD$GT_IDC_EVDLD_Set_Evnt    = $ASCIC (''),
;    0214  1  CRD$GT_IDC_EVDLD_Dev_Name       = $ASCIC ('LCA'),
;    0215  1 ! CRD$GT_IDC_EVDLD_Dev_Name       = $ASCIC ('TTA6'),
;    0216  1  CRD$GT_IDC_EVDLD_Strt_Qual  = $ASCIC (''),
;    0217  1  CRD$GT_IDC_EVDLD_Tst_Strt   = $ASCIC ('DMF32P         LCA'),
;    0218  1  CRD$GT_IDC_EVDLD_RunTime        = $ASCIC ('0:15'),
;    0219  1  CRD$GT_IDC_EVDLD_Prep_Err   = $ASCIC ('LCA: COMBO BOARD TESTING INCOMPLETE');
;    0220  1
```

```
;    0221  1
;    0222  1 %SBTTL 'Local Bindings for LESI/AZTEC based system'
;    0223  1
;    0224  1 BIND                ![04]
;    0225  1
;    0226  1  !+
;    0227  1  ! This data for EVRMB is for the RC25 (on DAA0) test.
;    0228  1  !-
;    0229  1
;    0230  1  CRD$GT_LESI_EVRMB_0    = $ASCIC ('EVRMB'),
;    0231  1  CRD$GT_LESI_EVRMB_0_Set_Flgs = $ASCIC ('QUICK'),
;    0232  1  CRD$GT_LESI_EVRMB_0_Set_Evnt = $ASCIC (''),
;    0233  1  CRD$GT_LESI_EVRMB_0_Dev_Name = $ASCIC ('DAA0'),
;    0234  1 ! CRD$GT_LESI_EVRMB_0_Dev_Name      = $ASCIC ('TTA6'),
;    0235  1  CRD$GT_LESI_EVRMB_0_Strt_Qual = $ASCIC ('/SECTION:CRD'),
;    0236  1  CRD$GT_LESI_EVRMB_0_Tst_Strt  = $ASCIC ('RC25 PART 2   DAA0'),
;    0237  1  CRD$GT_LESI_EVRMB_0_RunTime  = $ASCIC ('1:00'),
;    0238  1  CRD$GT_LESI_EVRMB_0_Prep_Err = $ASCIC ('DAA0: DISK DRIVE TESTING INCOMPLETE'),
;    0239  1
;    0240  1  !+
;    0241  1  ! This data for EVRMB is for the RCF25 (on DAA1) test.
;    0242  1  !-
;    0243  1
;    0244  1  CRD$GT_LESI_EVRMB_1    = $ASCIC ('EVRMB'),
;    0245  1  CRD$GT_LESI_EVRMB_1_Set_Flgs = $ASCIC ('QUICK'),
;    0246  1  CRD$GT_LESI_EVRMB_1_Set_Evnt = $ASCIC (''),
;    0247  1  CRD$GT_LESI_EVRMB_1_Dev_Name = $ASCIC ('DAA1'),
;    0248  1 ! CRD$GT_LESI_EVRMB_1_Dev_Name      = $ASCIC ('TTA6'),
;    0249  1  CRD$GT_LESI_EVRMB_1_Strt_Qual = $ASCIC ('/SECTION:CRD'),
;    0250  1  CRD$GT_LESI_EVRMB_1_Tst_Strt  = $ASCIC ('RCF25         DAA1'),
;    0251  1  CRD$GT_LESI_EVRMB_1_RunTime  = $ASCIC ('1:00'),
;    0252  1  CRD$GT_LESI_EVRMB_1_Prep_Err = $ASCIC ('DAA1: DISK DRIVE TESTING INCOMPLETE')
;    0253  1  ;
;    0254  1
;    0255  1
;    0256  1 ! !+
;    0257  1 ! ! This data for ENWCA is for the VS125 test.
;    0258  1 ! !-
;    0259  1 !
;    0260  1 ! CRD$GT_LESI_ENWCA    = $ASCIC ('ENWCA'),      ![05]
;    0261  1 ! CRD$GT_LESI_ENWCA_Set_Flgs = $ASCIC ('QUICK'),
;    0262  1 ! CRD$GT_LESI_ENWCA_Set_Evnt = $ASCIC (''),
;    0263  1 ! CRD$GT_LESI_ENWCA_Dev_Name = $ASCIC ('VBA0'),
;    0264  1 ! CRD$GT_LESI_ENWCA_Strt_Qual = $ASCIC ('/SECTION:MICRO'),
;    0265  1 ! CRD$GT_LESI_ENWCA_Tst_Strt  = $ASCIC ('VS125         VBA0'),
;    0266  1 ! CRD$GT_LESI_ENWCA_RunTime  = $ASCIC ('0:30'),
;    0267  1 ! CRD$GT_LESI_ENWCA_Prep_Err = $ASCIC ('VBA0: TERMINAL DRIVE TESTING INCOMPLETE');
;    0268  1
```

```
;    0269  1
;    0270  1
;    0271  1 %SBTTL 'Bindings for UDA based system, no RA60 Dr1, RA80 or 81 Dr0'
;    0272  1
;    0273  1 BIND                    ![04]
;    0274  1
;    0275  1  !+
;    0276  1  ! This data for EVRLA is for the RA-disk on drive 0
;    0277  1  ! Leave the Test_Start_Text device name field blank as a default. The field will be filled
;    0278  1  ! in later if the device can be attached by the AutoSizer.
;    0279  1  !-
;    0280  1
;    0281  1  CRD$GT_UDA_0_EVRLA_0    = $ASCIC ('EVRLA'),
;    0282  1  CRD$GT_UDA_0_EVRLA_0_Set_Flgs = $ASCIC ('QUICK'),
;    0283  1  CRD$GT_UDA_0_EVRLA_0_Set_Evnt = $ASCIC (''),
;    0284  1  CRD$GT_UDA_0_EVRLA_0_Dev_Name = $ASCIC ('DUA0'),
;    0285  1 ! CRD$GT_UDA_0_EVRLA_0_Dev_Name      = $ASCIC ('TTA6'),
;    0286  1  CRD$GT_UDA_0_EVRLA_0_Strt_Qual = $ASCIC ('/SECTION:CRD12'),
;    0287  1  CRD$GT_UDA_0_EVRLA_0_Tst_Strt  = $ASCIC ('     PART 2   DUA0'),
;    0288  1  CRD$GT_UDA_0_EVRLA_0_RunTime  = $ASCIC ('2:45'),
;    0289  1  CRD$GT_UDA_0_EVRLA_0_Prep_Err = $ASCIC ('DUA0: DISK DRIVE TESTING INCOMPLETE'),
;    0290  1
;    0291  1  !+
;    0292  1  ! This data for EVDLB is for the XGA0 (on the Combo board) test.
;    0293  1  !-
;    0294  1
;    0295  1  CRD$GT_UDA_0_EVDLB         = $ASCIC ('EVDLB'),
;    0296  1  CRD$GT_UDA_0_EVDLB_Set_Flgs    = $ASCIC (''),
;    0297  1  CRD$GT_UDA_0_EVDLB_Set_Evnt    = $ASCIC (''),
;    0298  1  CRD$GT_UDA_0_EVDLB_Dev_Name       = $ASCIC ('XGA0'),
;    0299  1 ! CRD$GT_UDA_0_EVDLB_Dev_Name       = $ASCIC ('TTA6'),
;    0300  1  CRD$GT_UDA_0_EVDLB_Strt_Qual  = $ASCIC (''),
;    0301  1  CRD$GT_UDA_0_EVDLB_Tst_Strt   = $ASCIC ('DMF32S        XGA0'),
;    0302  1  CRD$GT_UDA_0_EVDLB_RunTime       = $ASCIC ('1:00'),
;    0303  1  CRD$GT_UDA_0_EVDLB_Prep_Err   = $ASCIC ('XGA0: COMBO BOARD TESTING INCOMPLETE'),
;    0304  1
;    0305  1  !+
;    0306  1  ! This data for EVDLC is for the TXA (on the Combo board) test.
;    0307  1  !-
;    0308  1
;    0309  1  CRD$GT_UDA_0_EVDLC         = $ASCIC ('EVDLC'),
;    0310  1  CRD$GT_UDA_0_EVDLC_Set_Flgs    = $ASCIC ('QUICK'),
;    0311  1  CRD$GT_UDA_0_EVDLC_Set_Evnt    = $ASCIC (''),
;    0312  1  CRD$GT_UDA_0_EVDLC_Dev_Name       = $ASCIC ('TXA'),
;    0313  1 ! CRD$GT_UDA_0_EVDLC_Dev_Name       = $ASCIC ('TTA6'),
;    0314  1  CRD$GT_UDA_0_EVDLC_Strt_Qual  = $ASCIC (''),
;    0315  1  CRD$GT_UDA_0_EVDLC_Tst_Strt   = $ASCIC ('DMF32A   TXA0-TXA7'),
;    0316  1  CRD$GT_UDA_0_EVDLC_RunTime       = $ASCIC ('0:30'),
;    0317  1  CRD$GT_UDA_0_EVDLC_Prep_Err   = $ASCIC ('TXA: COMBO BOARD TESTING INCOMPLETE'),
;    0318  1
;    0319  1  !+
;    0320  1  ! This data for EVDLD is for the LCA (on the Combo board) test.
;    0321  1  !-
;    0322  1
;    0323  1  CRD$GT_UDA_0_EVDLD         = $ASCIC ('EVDLD'),
```

```
;    0324  1  CRD$GT_UDA_0_EVDLD_Set_Flgs     = $ASCIC (''),
;    0325  1  CRD$GT_UDA_0_EVDLD_Set_Evnt     = $ASCIC (''),
;    0326  1  CRD$GT_UDA_0_EVDLD_Dev_Name        = $ASCIC ('LCA'),
;    0327  1 ! CRD$GT_UDA_0_EVDLD_Dev_Name        = $ASCIC ('TTA6'),
;    0328  1  CRD$GT_UDA_0_EVDLD_Strt_Qual   = $ASCIC (''),
;    0329  1  CRD$GT_UDA_0_EVDLD_Tst_Strt    = $ASCIC ('DMF32P         LCA'),
;    0330  1  CRD$GT_UDA_0_EVDLD_RunTime        = $ASCIC ('0:15'),
;    0331  1  CRD$GT_UDA_0_EVDLD_Prep_Err    = $ASCIC ('LCA: COMBO BOARD TESTING INCOMPLETE');
;    0332  1
;    0333  1
;    0334  1
```

```
;    0335  1
;    0336  1 %SBTTL 'Bindings for UDA based system, RA60 Dr1, RA80 or 81 Dr0'
;    0337  1
;    0338  1 BIND                ![04]
;    0339  1
;    0340  1  !+
;    0341  1  ! This data for EVRLA is for the RA-disk on Drive 0 test.
;    0342  1  ! Leave the Test_Start_Text device name field blank as a default. The field will be filled
;    0343  1  ! in later if the device can be attached by the AutoSizer.
;    0344  1  !-
;    0345  1
;    0346  1  CRD$GT_UDA_1_EVRLA_0    = $ASCIC ('EVRLA'),
;    0347  1  CRD$GT_UDA_1_EVRLA_0_Set_Flgs = $ASCIC ('QUICK'),
;    0348  1  CRD$GT_UDA_1_EVRLA_0_Set_Evnt = $ASCIC (''),
;    0349  1  CRD$GT_UDA_1_EVRLA_0_Dev_Name = $ASCIC ('DUA0'),
;    0350  1 ! CRD$GT_UDA_1_EVRLA_0_Dev_Name      = $ASCIC ('TTA6'),
;    0351  1  CRD$GT_UDA_1_EVRLA_0_Strt_Qual = $ASCIC ('/SECTION:CRD12'),
;    0352  1  CRD$GT_UDA_1_EVRLA_0_Tst_Strt  = $ASCIC ('                DUA0'),
;    0353  1  CRD$GT_UDA_1_EVRLA_0_RunTime  = $ASCIC ('2:45'),
;    0354  1  CRD$GT_UDA_1_EVRLA_0_Prep_Err = $ASCIC ('DUA0: DISK DRIVE TESTING INCOMPLETE'),
;    0355  1
;    0356  1  !+
;    0357  1  ! This data for EVRLA is for the Drive 1 test.
;    0358  1  !-
;    0359  1
;    0360  1  CRD$GT_UDA_1_EVRLA_1    = $ASCIC ('EVRLA'),
;    0361  1  CRD$GT_UDA_1_EVRLA_1_Set_Flgs = $ASCIC ('QUICK'),
;    0362  1  CRD$GT_UDA_1_EVRLA_1_Set_Evnt = $ASCIC (''),
;    0363  1  CRD$GT_UDA_1_EVRLA_1_Dev_Name = $ASCIC ('DJA1'),
;    0364  1 ! CRD$GT_UDA_1_EVRLA_1_Dev_Name = $ASCIC ('TTA6'),
;    0365  1  CRD$GT_UDA_1_EVRLA_1_Strt_Qual = $ASCIC ('/SECTION:CRD12'),
;    0366  1  CRD$GT_UDA_1_EVRLA_1_Tst_Strt = $ASCIC ('RA60 PART 2   DJA1'),
;    0367  1  CRD$GT_UDA_1_EVRLA_1_RunTime  = $ASCIC ('1:00'),
;    0368  1  CRD$GT_UDA_1_EVRLA_1_Prep_Err = $ASCIC ('DJA1: DISK DRIVE TESTING INCOMPLETE'),
;    0369  1
;    0370  1  !+
;    0371  1  ! This data for EVDLB is for the XGA0 (on the Combo board) test.
;    0372  1  !-
;    0373  1
;    0374  1  CRD$GT_UDA_1_EVDLB          = $ASCIC ('EVDLB'),
;    0375  1  CRD$GT_UDA_1_EVDLB_Set_Flgs    = $ASCIC (''),
;    0376  1  CRD$GT_UDA_1_EVDLB_Set_Evnt    = $ASCIC (''),
;    0377  1  CRD$GT_UDA_1_EVDLB_Dev_Name       = $ASCIC ('XGA0'),
;    0378  1 ! CRD$GT_UDA_1_EVDLB_Dev_Name       = $ASCIC ('TTA6'),
;    0379  1  CRD$GT_UDA_1_EVDLB_Strt_Qual  = $ASCIC (''),
;    0380  1  CRD$GT_UDA_1_EVDLB_Tst_Strt   = $ASCIC ('DMF32S        XGA0'),
;    0381  1  CRD$GT_UDA_1_EVDLB_RunTime       = $ASCIC ('1:00'),
;    0382  1  CRD$GT_UDA_1_EVDLB_Prep_Err   = $ASCIC ('XGA0: COMBO BOARD TESTING INCOMPLETE'),
;    0383  1
;    0384  1  !+
;    0385  1  ! This data for EVDLC is for the TXA (on the Combo board) test.
;    0386  1  !-
;    0387  1
;    0388  1  CRD$GT_UDA_1_EVDLC          = $ASCIC ('EVDLC'),
;    0389  1  CRD$GT_UDA_1_EVDLC_Set_Flgs    = $ASCIC ('QUICK'),
```

```
;    0390  1  CRD$GT_UDA_1_EVDLC_Set_Evnt    = $ASCIC (''),
;    0391  1  CRD$GT_UDA_1_EVDLC_Dev_Name      = $ASCIC ('TXA'),
;    0392  1 ! CRD$GT_UDA_1_EVDLC_Dev_Name      = $ASCIC ('TTA6'),
;    0393  1  CRD$GT_UDA_1_EVDLC_Strt_Qual   = $ASCIC (''),
;    0394  1  CRD$GT_UDA_1_EVDLC_Tst_Strt    = $ASCIC ('DMF32A   TXA0-TXA7'),
;    0395  1  CRD$GT_UDA_1_EVDLC_RunTime      = $ASCIC ('0:30'),
;    0396  1  CRD$GT_UDA_1_EVDLC_Prep_Err    = $ASCIC ('TXA: COMBO BOARD TESTING INCOMPLETE'),
;    0397  1
;    0398  1  !+
;    0399  1  ! This data for EVDLD is for the LCA (on the Combo board) test.
;    0400  1  !-
;    0401  1
;    0402  1  CRD$GT_UDA_1_EVDLD          = $ASCIC ('EVDLD'),
;    0403  1  CRD$GT_UDA_1_EVDLD_Set_Flgs    = $ASCIC (''),
;    0404  1  CRD$GT_UDA_1_EVDLD_Set_Evnt    = $ASCIC (''),
;    0405  1  CRD$GT_UDA_1_EVDLD_Dev_Name      = $ASCIC ('LCA'),
;    0406  1 ! CRD$GT_UDA_1_EVDLD_Dev_Name      = $ASCIC ('TTA6'),
;    0407  1  CRD$GT_UDA_1_EVDLD_Strt_Qual   = $ASCIC (''),
;    0408  1  CRD$GT_UDA_1_EVDLD_Tst_Strt    = $ASCIC ('DMF32P         LCA'),
;    0409  1  CRD$GT_UDA_1_EVDLD_RunTime      = $ASCIC ('0:15'),
;    0410  1  CRD$GT_UDA_1_EVDLD_Prep_Err    = $ASCIC ('LCA: COMBO BOARD TESTING INCOMPLETE');
;    0411  1
```

```
;    0412  1 %SBTTL 'Bindings for UDA based system, no RA60 Dr1, RA60 Dr0'
;    0413  1
;    0414  1 BIND                ![04]
;    0415  1
;    0416  1  !+
;    0417  1  ! This data for EVRLA is for the RA-disk on drive 0
;    0418  1  ! Leave the Test_Start_Text device name field blank as a default. The field will be filled
;    0419  1  ! in later if the device can be attached by the AutoSizer.
;    0420  1  !-
;    0421  1
;    0422  1  CRD$GT_UDA_2_EVRLA_0    = $ASCIC ('EVRLA'),
;    0423  1  CRD$GT_UDA_2_EVRLA_0_Set_Flgs = $ASCIC ('QUICK'),
;    0424  1  CRD$GT_UDA_2_EVRLA_0_Set_Evnt = $ASCIC (''),
;    0425  1  CRD$GT_UDA_2_EVRLA_0_Dev_Name = $ASCIC ('DJA0'),
;    0426  1 ! CRD$GT_UDA_2_EVRLA_0_Dev_Name       = $ASCIC ('TTA6'),
;    0427  1  CRD$GT_UDA_2_EVRLA_0_Strt_Qual = $ASCIC ('/SECTION:CRD12'),
;    0428  1  CRD$GT_UDA_2_EVRLA_0_Tst_Strt  = $ASCIC ('RA60 PART 2   DJA0'),
;    0429  1  CRD$GT_UDA_2_EVRLA_0_RunTime  = $ASCIC ('1:00'),
;    0430  1  CRD$GT_UDA_2_EVRLA_0_Prep_Err = $ASCIC ('DJA0: DISK DRIVE TESTING INCOMPLETE'),
;    0431  1
;    0432  1  !+
;    0433  1  ! This data for EVDLB is for the XGA0 (on the Combo board) test.
;    0434  1  !-
;    0435  1
;    0436  1  CRD$GT_UDA_2_EVDLB          = $ASCIC ('EVDLB'),
;    0437  1  CRD$GT_UDA_2_EVDLB_Set_Flgs    = $ASCIC (''),
;    0438  1  CRD$GT_UDA_2_EVDLB_Set_Evnt    = $ASCIC (''),
;    0439  1  CRD$GT_UDA_2_EVDLB_Dev_Name       = $ASCIC ('XGA0'),
;    0440  1 ! CRD$GT_UDA_2_EVDLB_Dev_Name       = $ASCIC ('TTA6'),
;    0441  1  CRD$GT_UDA_2_EVDLB_Strt_Qual  = $ASCIC (''),
;    0442  1  CRD$GT_UDA_2_EVDLB_Tst_Strt   = $ASCIC ('DMF32S        XGA0'),
;    0443  1  CRD$GT_UDA_2_EVDLB_RunTime       = $ASCIC ('1:00'),
;    0444  1  CRD$GT_UDA_2_EVDLB_Prep_Err   = $ASCIC ('XGA0: COMBO BOARD TESTING INCOMPLETE'),
;    0445  1
;    0446  1  !+
;    0447  1  ! This data for EVDLC is for the TXA (on the Combo board) test.
;    0448  1  !-
;    0449  1
;    0450  1  CRD$GT_UDA_2_EVDLC          = $ASCIC ('EVDLC'),
;    0451  1  CRD$GT_UDA_2_EVDLC_Set_Flgs    = $ASCIC ('QUICK'),
;    0452  1  CRD$GT_UDA_2_EVDLC_Set_Evnt    = $ASCIC (''),
;    0453  1  CRD$GT_UDA_2_EVDLC_Dev_Name       = $ASCIC ('TXA'),
;    0454  1 ! CRD$GT_UDA_2_EVDLC_Dev_Name       = $ASCIC ('TTA6'),
;    0455  1  CRD$GT_UDA_2_EVDLC_Strt_Qual  = $ASCIC (''),
;    0456  1  CRD$GT_UDA_2_EVDLC_Tst_Strt   = $ASCIC ('DMF32A   TXA0-TXA7'),
;    0457  1  CRD$GT_UDA_2_EVDLC_RunTime       = $ASCIC ('0:30'),
;    0458  1  CRD$GT_UDA_2_EVDLC_Prep_Err   - $ASCIC ('TXA: COMBO BOARD TESTING INCOMPLETE'),
;    0459  1
;    0460  1  !+
;    0461  1  ! This data for EVDLD is for the LCA (on the Combo board) test.
;    0462  1  !-
;    0463  1
;    0464  1  CRD$GT_UDA_2_EVDLD          - $ASCIC ('EVDLD'),
;    0465  1  CRD$GT_UDA_2_EVDLD_Set_Flgs    = $ASCIC (''),
;    0466  1  CRD$GT_UDA_2_EVDLD_Set_Evnt    - $ASCIC (''),
```

```
;    0467  1  CRD$GT_UDA_2_EVDLD_Dev_Name       = $ASCIC ('LCA'),
;    0468  1 ! CRD$GT_UDA_2_EVDLD_Dev_Name       = $ASCIC ('TTA6'),
;    0469  1  CRD$GT_UDA_2_EVDLD_Strt_Qual   = $ASCIC (''),
;    0470  1  CRD$GT_UDA_2_EVDLD_Tst_Strt    = $ASCIC ('DMF32P         LCA'),
;    0471  1  CRD$GT_UDA_2_EVDLD_RunTime        = $ASCIC ('0:15'),
;    0472  1  CRD$GT_UDA_2_EVDLD_Prep_Err    = $ASCIC ('LCA: COMBO BOARD TESTING INCOMPLETE');
;    0473  1
;    0474  1
;    0475  1
```

```
;    0476  1 %SBTTL 'Bindings for UDA based system, RA60 Dr1, RA60 Dr0'
;    0477  1
;    0478  1 BIND                ![04]
;    0479  1
;    0480  1  !+
;    0481  1  ! This data for EVRLA is for the RA-disk on Drive 0 test.
;    0482  1  ! Leave the Test_Start_Text device name field blank as a default. The field will be filled
;    0483  1  ! in later if the device can be attached by the AutoSizer.
;    0484  1  !-
;    0485  1
;    0486  1  CRD$GT_UDA_3_EVRLA_0    = $ASCIC ('EVRLA'),
;    0487  1  CRD$GT_UDA_3_EVRLA_0_Set_Flgs = $ASCIC ('QUICK'),
;    0488  1  CRD$GT_UDA_3_EVRLA_0_Set_Evnt = $ASCIC (''),
;    0489  1  CRD$GT_UDA_3_EVRLA_0_Dev_Name = $ASCIC ('DJA0'),
;    0490  1 ! CRD$GT_UDA_3_EVRLA_0_Dev_Name      = $ASCIC ('TTA6'),
;    0491  1  CRD$GT_UDA_3_EVRLA_0_Strt_Qual = $ASCIC ('/SECTION:CRD12'),
;    0492  1  CRD$GT_UDA_3_EVRLA_0_Tst_Strt  = $ASCIC ('                DJA0'),
;    0493  1  CRD$GT_UDA_3_EVRLA_0_RunTime  = $ASCIC ('1:00'),
;    0494  1  CRD$GT_UDA_3_EVRLA_0_Prep_Err = $ASCIC ('DJA0: DISK DRIVE TESTING INCOMPLETE'),
;    0495  1
;    0496  1  !+
;    0497  1  ! This data for EVRLA is for the Drive 1 test.
;    0498  1  !-
;    0499  1
;    0500  1  CRD$GT_UDA_3_EVRLA_1    = $ASCIC ('EVRLA'),
;    0501  1  CRD$GT_UDA_3_EVRLA_1_Set_Flgs = $ASCIC ('QUICK'),
;    0502  1  CRD$GT_UDA_3_EVRLA_1_Set_Evnt = $ASCIC (''),
;    0503  1  CRD$GT_UDA_3_EVRLA_1_Dev_Name = $ASCIC ('DJA1'),
;    0504  1 ! CRD$GT_UDA_3_EVRLA_1_Dev_Name = $ASCIC ('TTA6'),
;    0505  1  CRD$GT_UDA_3_EVRLA_1_Strt_Qual = $ASCIC ('/SECTION:CRD12'),
;    0506  1  CRD$GT_UDA_3_EVRLA_1_Tst_Strt = $ASCIC ('RA60 PART 2   DJA1'),
;    0507  1  CRD$GT_UDA_3_EVRLA_1_RunTime  = $ASCIC ('1:00'),
;    0508  1  CRD$GT_UDA_3_EVRLA_1_Prep_Err = $ASCIC ('DJA1: DISK DRIVE TESTING INCOMPLETE'),
;    0509  1
;    0510  1  !+
;    0511  1  ! This data for EVDLB is for the XGA0 (on the Combo board) test.
;    0512  1  !-
;    0513  1
;    0514  1  CRD$GT_UDA_3_EVDLB          = $ASCIC ('EVDLB'),
;    0515  1  CRD$GT_UDA_3_EVDLB_Set_Flgs     = $ASCIC (''),
;    0516  1  CRD$GT_UDA_3_EVDLB_Set_Evnt     = $ASCIC (''),
;    0517  1  CRD$GT_UDA_3_EVDLB_Dev_Name       = $ASCIC ('XGA0'),
;    0518  1 ! CRD$GT_UDA_3_EVDLB_Dev_Name       = $ASCIC ('TTA6'),
;    0519  1  CRD$GT_UDA_3_EVDLB_Strt_Qual   = $ASCIC (''),
;    0520  1  CRD$GT_UDA_3_EVDLB_Tst_Strt    = $ASCIC ('DMF32S        XGA0'),
;    0521  1  CRD$GT_UDA_3_EVDLB_RunTime       = $ASCIC ('1:00'),
;    0522  1  CRD$GT_UDA_3_EVDLB_Prep_Err    = $ASCIC ('XGA0: COMBO BOARD TESTING INCOMPLETE'),
;    0523  1
;    0524  1  !+
;    0525  1  ! This data for EVDLC is for the TXA (on the Combo board) test.
;    0526  1  !-
;    0527  1
;    0528  1  CRD$GT_UDA_3_EVDLC          = $ASCIC ('EVDLC'),
;    0529  1  CRD$GT_UDA_3_EVDLC_Set_Flgs     = $ASCIC ('QUICK'),
;    0530  1  CRD$GT_UDA_3_EVDLC_Set_Evnt     - $ASCIC (''),
```

```
;    0531  1  CRD$GT_UDA_3_EVDLC_Dev_Name       = $ASCIC ('TXA'),
;    0532  1 ! CRD$GT_UDA_3_EVDLC_Dev_Name       = $ASCIC ('TTA6'),
;    0533  1  CRD$GT_UDA_3_EVDLC_Strt_Qual   = $ASCIC (''),
;    0534  1  CRD$GT_UDA_3_EVDLC_Tst_Strt    = $ASCIC ('DMF32A   TXA0-TXA7'),
;    0535  1  CRD$GT_UDA_3_EVDLC_RunTime       = $ASCIC ('0:30'),
;    0536  1  CRD$GT_UDA_3_EVDLC_Prep_Err    = $ASCIC ('TXA: COMBO BOARD TESTING INCOMPLETE'),
;    0537  1
;    0538  1  !+
;    0539  1  ! This data for EVDLD is for the LCA (on the Combo board) test.
;    0540  1  !-
;    0541  1
;    0542  1  CRD$GT_UDA_3_EVDLD          = $ASCIC ('EVDLD'),
;    0543  1  CRD$GT_UDA_3_EVDLD_Set_Flgs    = $ASCIC (''),
;    0544  1  CRD$GT_UDA_3_EVDLD_Set_Evnt    = $ASCIC (''),
;    0545  1  CRD$GT_UDA_3_EVDLD_Dev_Name       = $ASCIC ('LCA'),
;    0546  1 ! CRD$GT_UDA_3_EVDLD_Dev_Name       = $ASCIC ('TTA6'),
;    0547  1  CRD$GT_UDA_3_EVDLD_Strt_Qual   = $ASCIC (''),
;    0548  1  CRD$GT_UDA_3_EVDLD_Tst_Strt    = $ASCIC ('DMF32P         LCA'),
;    0549  1  CRD$GT_UDA_3_EVDLD_RunTime       = $ASCIC ('0:15'),
;    0550  1  CRD$GT_UDA_3_EVDLD_Prep_Err    = $ASCIC ('LCA: COMBO BOARD TESTING INCOMPLETE');
;    0551  1
```

```
;   0552  1
```

```
;    0553  1 %SBTTL 'Global Bindings'
;    0554  1
;    0555  1 GLOBAL BIND
;    0556  1  CRD$GT_DS_LOAD_Com    = $ASCIC ('LOAD '),
;    0557  1  CRD$GT_DS_SET_FLAGS_Com    = $ASCIC ('SET FLAGS '),
;    0558  1  CRD$GT_DS_SET_EVENT_Com    = $ASCIC ('SET EVENT FLAGS '),
;    0559  1  CRD$GT_DS_SELECT_Com    = $ASCIC ('SELECT '),
;    0560  1  CRD$GT_DS_START_Com    = $ASCIC ('START ');
```

```
;    0561  1 %SBTTL 'CRD$Build_Support_Table'
;    0562  1
;    0563  1 GLOBAL ROUTINE CRD$Build_Support_Table : NOVALUE =
;    0564  1
;    0565  1 !++
;    0566  1 ! Functional Description:
;    0567  1 !
;    0568  1 ! Initialize all Hardware_Support tables.        [04]
;    0569  1 !
;    0570  1 ! Formal Input Parameters:
;    0571  1 !
;    0572  1 ! None
;    0573  1 !
;    0574  1 ! Formal Output Parameters:
;    0575  1 !
;    0576  1 ! None
;    0577  1 !
;    0578  1 ! Side Effects:
;    0579  1 !
;    0580  1 ! None
;    0581  1 !
;    0582  1 ! Completion Codes:
;    0583  1 !
;    0584  1 ! None
;    0585  1 -
```

```
;   0586  2 BEGIN     ! Routine CRD$Build_Support_Table
;   0587  2  !+
;   0588  2  ! As usual, define some macros to help build the table. Aren't macros wonderful?   [04]
;   0589  2  !-
;   0590  2
;   0591  2  MACRO
; M 0592  2    Insert_Diagnostic_Name (Diag,System) =
; M 0593  2     (
; M 0594  2 !    EXTERNAL
; M 0595  2 !     %NAME ('CRD$GT_', System, '_', Diag) : ADDRESSING_MODE (LONG_RELATIVE);
; M 0596  2
; M 0597  2     %NAME ('CRD$AL_',System,'_Hdwr_Sprt_Table') [%NAME ('CRD$K_', System, '_', Diag), CRD$K_Diagnostic_Name]
; M 0598  2      %NAME ('CRD$GT_', System, '_', Diag);
; M 0599  2     )
;   0600  2    %,
;   0601  2
; M 0602  2    Insert_Set_Flgs (Diag,System) =
; M 0603  2     (
; M 0604  2 !    EXTERNAL
; M 0605  2 !     %NAME ('CRD$GT_', System, '_', Diag) : ADDRESSING_MODE (LONG_RELATIVE);
; M 0606  2
; M 0607  2     %NAME ('CRD$AL_',System,'_Hdwr_Sprt_Table') [%NAME ('CRD$K_', System, '_', Diag), CRD$K_Set_Flags_Args] -
; M 0608  2      %NAME ('CRD$GT_', System, '_', Diag, '_Set_Flgs');
; M 0609  2     )
;   0610  2    %,
;   0611  2
; M 0612  2    Insert_Set_Evnt (Diag,System) =
; M 0613  2     (
; M 0614  2 !    EXTERNAL
; M 0615  2 !     %NAME ('CRD$GT_', System, '_', Diag) : ADDRESSING_MODE (LONG_RELATIVE);
; M 0616  2
; M 0617  2     %NAME ('CRD$AL_',System,'_Hdwr_Sprt_Table') [%NAME ('CRD$K_', System, '_', Diag), CRD$K_Set_Event_Args] =
; M 0618  2      %NAME ('CRD$GT_', System, '_', Diag, '_Set_Evnt');
; M 0619  2     )
;   0620  2    %,
;   0621  2
; M 0622  2    Insert_Strt_Qual (Diag,System) =
; M 0623  2     (
; M 0624  2 !    EXTERNAL
; M 0625  2 !     %NAME ('CRD$GT_', System, '_', Diag) : ADDRESSING_MODE (LONG_RELATIVE);
; M 0626  2
; M 0627  2     %NAME ('CRD$AL_',System,'_Hdwr_Sprt_Table') [%NAME ('CRD$K_', System, '_', Diag), CRD$K_Start_Qualifiers] =
; M 0628  2      %NAME ('CRD$GT_', System, '_', Diag, '_Strt_Qual');
; M 0629  2     )
;   0630  2    %,
```

```
; M 0631  2    Insert_Dev_Name (Diag,System) =
; M 0632  2     (
; M 0633  2 !    EXTERNAL
; M 0634  2 !     %NAME ('CRD$GT_', System, '_', Diag) : ADDRESSING_MODE (LONG_RELATIVE);
; M 0635  2
; M 0636  2     %NAME ('CRD$AL_', System, '_Hdwr_Sprt_Table') [%NAME ('CRD$K_', System, '_', Diag), CRD$K_Device_Name] -
; M 0637  2      %NAME ('CRD$GT_', System, '_', Diag, '_Dev_Name');
; M 0638  2     )
;   0639  2    %,
;   0640  2
; M 0641  2    Insert_RunTime (Diag,System) =
; M 0642  2     (
; M 0643  2 !    EXTERNAL
; M 0644  2 !     %NAME ('CRD$GT_', System, '_', Diag) : ADDRESSING_MODE (LONG_RELATIVE);
; M 0645  2
; M 0646  2     %NAME ('CRD$AL_', System, '_Hdwr_Sprt_Table') [%NAME ('CRD$K_', System, '_', Diag), CRD$K_RunTime] =
; M 0647  2      %NAME ('CRD$GT_', System, '_', Diag, '_RunTime');
; M 0648  2     )
;   0649  2    %,
;   0650  2
; M 0651  2    Insert_Prep_Err (Diag,System) =
; M 0652  2     (
; M 0653  2 !    EXTERNAL
; M 0654  2 !     %NAME ('CRD$GT_', System, '_', Diag) : ADDRESSING_MODE (LONG_RELATIVE);
; M 0655  2
; M 0656  2     %NAME ('CRD$AL_', System, '_Hdwr_Sprt_Table') [%NAME ('CRD$K_', System, '_', Diag), CRD$K_Preparation_Error_Text] =
; M 0657  2      %NAME ('CRD$GT_', System, '_', Diag, '_Prep_Err');
; M 0658  2     )
;   0659  2    %,
;   0660  2
; M 0661  2    Insert_Tst_Strt (Diag,System) =
; M 0662  2     (
; M 0663  2 !    EXTERNAL
; M 0664  2 !     %NAME ('CRD$GT_', System, '_', Diag) : ADDRESSING_MODE (LONG_RELATIVE);
; M 0665  2
; M 0666  2     %NAME ('CRD$AL_', System, '_Hdwr_Sprt_Table') [%NAME ('CRD$K_', System, '_', Diag), CRD$K_Test_Start_Text] =
; M 0667  2      %NAME ('CRD$GT_', System, '_', Diag, '_Tst_Strt');
; M 0668  2     )
;   0669  2    %;
```

```
;    0670  2 !+
;    0671  2 ! Initialize Hardware Support Table for IDC based system        [04]
;    0672  2 !-
;    0673  2
;    0674  2  !
;    0675  2  ! Initialize the IDC Hardware Support Table to all zero pointers.
;    0676  2  !
;    0677  2
;    0678  2  INCR Diag FROM 0 TO (CRD$K_IDC_Last_Diagnostic - 1) DO
;    0679  2   INCR Entry FROM 0 TO (CRD$K_Last_Entry - 1) DO
;    0680  3    BEGIN
;    0681  3    CRD$AL_IDC_Hdwr_Sprt_Table [.Diag, .Entry] = 0;
;    0682  2    END;
;    0683  2
;    0684  2  !
;    0685  2  ! Insert pointers to the ASCIC strings into the IDC Hardware Support Table
;    0686  2  !
;    0687  2
;    0688  2  Insert_Diagnostic_Name (EVRAD_0, IDC);
;    0689  2  Insert_Set_Flgs (EVRAD_0, IDC);
;    0690  2  Insert_Set_Evnt (EVRAD_0, IDC);
;    0691  2  Insert_Strt_Qual (EVRAD_0, IDC);
;    0692  2  Insert_Dev_Name (EVRAD_0, IDC);
;    0693  2  Insert_Tst_Strt (EVRAD_0, IDC);
;    0694  2  Insert_RunTime  (EVRAD_0, IDC);
;    0695  2  Insert_Prep_Err (EVRAD_0, IDC);
;    0696  2
;    0697  2  Insert_Diagnostic_Name (EVRAD_1, IDC);
;    0698  2  Insert_Set_Flgs (EVRAD_1, IDC);
;    0699  2  Insert_Set_Evnt (EVRAD_1, IDC);
;    0700  2  Insert_Strt_Qual (EVRAD_1, IDC);
;    0701  2  Insert_Dev_Name (EVRAD_1, IDC);
;    0702  2  Insert_Tst_Strt (EVRAD_1, IDC);
;    0703  2  Insert_RunTime  (EVRAD_1, IDC);
;    0704  2  Insert_Prep_Err (EVRAD_1, IDC);
;    0705  2
;    0706  2  Insert_Diagnostic_Name (EVDLB, IDC);
;    0707  2  Insert_Set_Flgs (EVDLB, IDC);
;    0708  2  Insert_Set_Evnt (EVDLB, IDC);
;    0709  2  Insert_Strt_Qual (EVDLB, IDC);
;    0710  2  Insert_Dev_Name (EVDLB, IDC);
;    0711  2  Insert_Tst_Strt (EVDLB, IDC);
;    0712  2  Insert_RunTime  (EVDLB, IDC);
;    0713  2  Insert_Prep Err (EVDLB, IDC);
;    0714  2
;    0715  2  Insert_Diagnostic_Name (EVDLC, IDC);
;    0716  2  Insert_Set_Flgs (EVDLC, IDC);
;    0717  2  Insert_Set_Evnt (EVDLC, IDC);
;    0718  2  Insert_Strt_Qual (EVDLC, IDC);
;    0719  2  Insert_Dev_Name (EVDLC, IDC);
;    0720  2  Insert_Tst_Strt (EVDLC, IDC);
;    0721  2  Insert_RunTime  (EVDLC, IDC);
;    0722  2  Insert_Prep_Err (EVDLC, IDC);
;    0723  2
;    0724  2  Insert_Diagnostic_Name (EVDLD, IDC);
```

```
;    0725  2  Insert_Set_Flgs (EVDLD, IDC);
;    0726  2  Insert_Set_Evnt (EVDLD, IDC);
;    0727  2  Insert_Strt_Qual (EVDLD, IDC);
;    0728  2  Insert_Dev_Name (EVDLD, IDC);
;    0729  2  Insert_Tst_Strt (EVDLD, IDC);
;    0730  2  Insert_RunTime  (EVDLD, IDC);
;    0731  2  Insert_Prep_Err (EVDLD, IDC);
;    0732  2
;    0733  2
```

```
;    0734  2
;    0735  2 !+
;    0736  2 ! Initialize Hardware Support Table for LESI/AZTEC based system       [04]
;    0737  2 !-
;    0738  2
;    0739  2  !
;    0740  2  ! Initialize the LESI Hardware Support Table to all zero pointers.
;    0741  2  !
;    0742  2
;    0743  2  INCR Diag FROM 0 TO (CRD$K_LESI_Last_Diagnostic - 1) DO
;    0744  2   INCR Entry FROM 0 TO (CRD$K_Last_Entry - 1) DO
;    0745  3    BEGIN
;    0746  3    CRD$AL_LESI_Hdwr_Sprt_Table [.Diag, .Entry] = 0;
;    0747  2    END;
;    0748  2
;    0749  2  !
;    0750  2  ! Insert pointers to the ASCIC strings into the LESI Hardware Support Table
;    0751  2  !
;    0752  2
;    0753  2  Insert_Diagnostic_Name (EVRMB_0, LESI);
;    0754  2  Insert_Set_Flgs (EVRMB_0, LESI);
;    0755  2  Insert_Set_Evnt (EVRMB_0, LESI);
;    0756  2  Insert_Strt_Qual (EVRMB_0, LESI);
;    0757  2  Insert_Dev_Name (EVRMB_0, LESI);
;    0758  2  Insert_Tst_Strt (EVRMB_0, LESI);
;    0759  2  Insert_RunTime  (EVRMB_0, LESI);
;    0760  2  Insert_Prep_Err (EVRMB_0, LESI);
;    0761  2
;    0762  2  Insert_Diagnostic_Name (EVRMB_1, LESI);
;    0763  2  Insert_Set_Flgs (EVRMB_1, LESI);
;    0764  2  Insert_Set_Evnt (EVRMB_1, LESI);
;    0765  2  Insert_Strt_Qual (EVRMB_1, LESI);
;    0766  2  Insert_Dev_Name (EVRMB_1, LESI);
;    0767  2  Insert_Tst_Strt (EVRMB_1, LESI);
;    0768  2  Insert RunTime  (EVRMB_1, LESI);
;    0769  2  Insert_Prep_Err (EVRMB_1, LESI);
;    0770  2
;    0771  2
;    0772  2 ! Insert_Diagnostic_Name (ENWCA, LESI);        ![05]
;    0773  2 ! Insert_Set_Flgs (ENWCA, LESI);        ![05]
;    0774  2 ! Insert_Set_Evnt (ENWCA, LESI);        ![05]
;    0775  2 ! Insert_Strt_Qual (ENWCA, LESI);        ![05]        ![05]
;    0776  2 ! Insert_Dev_Name (ENWCA, LESI);        ![05]
;    0777  2 ! Insert_Tst_Strt (ENWCA, LESI);        ![05]
;    0778  2 ! Insert_RunTime  (ENWCA, LESI);        ![05]
;    0779  2 . Insert_Prep_Err (ENWCA, LESI);        .[05]
;    0780  2
;    0781  2
;    0782  2
```

```
;    0783  2
;    0784  2 !+
;    0785  2 ! Initialize Hardware Support Table for UDA based system with no RA60 for Drive 1    [04]
;    0786  2 ! and RA80 or 81 for Drive 0
;    0787  2 !-
;    0788  2
;    0789  2  !
;    0790  2  ! Initialize the UDA_0 Hardware Support Table to all zero pointers.
;    0791  2  !
;    0792  2
;    0793  2  INCR Diag FROM 0 TO (CRD$K_UDA_0_Last_Diagnostic - 1) DO
;    0794  2   INCR Entry FROM 0 TO (CRD$K_Last_Entry - 1) DO
;    0795  3    BEGIN
;    0796  3    CRD$AL_UDA_0_Hdwr_Sprt_Table [.Diag, .Entry] = 0;
;    0797  2    END;
;    0798  2
;    0799  2  !
;    0800  2  ! Insert pointers to the ASCIC strings into the UDA_0 Hardware Support Table
;    0801  2  !
;    0802  2
;    0803  2  Insert_Diagnostic_Name (EVRLA_0, UDA_0);
;    0804  2  Insert_Set_Flgs (EVRLA_0, UDA_0);
;    0805  2  Insert_Set_Evnt (EVRLA_0, UDA_0);
;    0806  2  Insert_Strt_Qual (EVRLA_0, UDA_0);
;    0807  2  Insert_Dev_Name (EVRLA_0, UDA_0);
;    0808  2  Insert_Tst_Strt (EVRLA_0, UDA_0);
;    0809  2  Insert_RunTime  (EVRLA_0, UDA_0);
;    0810  2  Insert_Prep_Err (EVRLA_0, UDA_0);
;    0811  2
;    0812  2  Insert_Diagnostic_Name (EVDLB, UDA_0);
;    0813  2  Insert_Set_Flgs (EVDLB, UDA_0);
;    0814  2  Insert_Set_Evnt (EVDLB, UDA_0);
;    0815  2  Insert_Strt_Qual (EVDLB, UDA_0);
;    0816  2  Insert_Dev_Name (EVDLB, UDA_0);
;    0817  2  Insert_Tst_Strt (EVDLB, UDA_0);
;    0818  2  Insert_RunTime  (EVDLB, UDA_0);
;    0819  2  Insert_Prep_Err (EVDLB, UDA_0);
;    0820  2
;    0821  2  Insert_Diagnostic_Name (EVDLC, UDA_0);
;    0822  2  Insert_Set_Flgs (EVDLC, UDA_0);
;    0823  2  Insert_Set_Evnt (EVDLC, UDA_0);
;    0824  2  Insert_Strt_Qual (EVDLC, UDA_0);
;    0825  2  Insert_Dev_Name (EVDLC, UDA_0);
;    0826  2  Insert_Tst_Strt (EVDLC, UDA_0);
;    0827  2  Insert_RunTime  (EVDLC, UDA_0);
;    0828  2  Insert_Prep_Err (EVDLC, UDA_0);
;    0829  2
;    0830  2  Insert_Diagnostic_Name (EVDLD, UDA_0);
;    0831  2  Insert_Set_Flgs (EVDLD, UDA_0);
;    0832  2  Insert_Set_Evnt (EVDLD, UDA_0);
;    0833  2  Insert_Strt_Qual (EVDLD, UDA_0);
;    0834  2  Insert_Dev_Name (EVDLD, UDA_0);
;    0835  2  Insert_Tst_Strt (EVDLD, UDA_0);
;    0836  2  Insert_RunTime  (EVDLD, UDA_0);
;    0837  2  Insert_Prep_Err (EVDLD, UDA_0);
```

```
;    0838  2
;    0839  2
```

```
;    0840  2
;    0841  2
;    0842  2
;    0843  2 !+
;    0844  2 ! Initialize Hardware Support Table for UDA based system with RA60 for Drive 1     [04]
;    0845  2 ! and RA80 or 81 for Drive 0
;    0846  2 !-
;    0847  2
;    0848  2  !
;    0849  2  ! Initialize the UDA_1 Hardware Support Table to all zero pointers.
;    0850  2  !
;    0851  2
;    0852  2  INCR Diag FROM 0 TO (CRD$K_UDA_1_Last_Diagnostic - 1) DO
;    0853  2   INCR Entry FROM 0 TO (CRD$K_Last_Entry - 1) DO
;    0854  3    BEGIN
;    0855  3    CRD$AL_UDA_1_Hdwr_Sprt_Table [.Diag, .Entry] = 0;
;    0856  2    END;
;    0857  2
;    0858  2  !
;    0859  2  ! Insert pointers to the ASCIC strings into the UDA_1 Hardware Support Table
;    0860  2  !
;    0861  2
;    0862  2  Insert_Diagnostic_Name (EVRLA_0, UDA_1);
;    0863  2  Insert_Set_Flgs (EVRLA_0, UDA_1);
;    0864  2  Insert_Set_Evnt (EVRLA_0, UDA_1);
;    0865  2  Insert_Strt_Qual (EVRLA_0, UDA_1);
;    0866  2  Insert_Dev_Name (EVRLA_0, UDA_1);
;    0867  2  Insert_Tst_Strt (EVRLA_0, UDA_1);
;    0868  2  Insert_RunTime  (EVRLA_0, UDA_1);
;    0869  2  Insert_Prep_Err (EVRLA_0, UDA_1);
;    0870  2
;    0871  2  Insert_Diagnostic_Name (EVRLA_1, UDA_1);
;    0872  2  Insert_Set_Flgs (EVRLA_1, UDA_1);
;    0873  2  Insert_Set_Evnt (EVRLA_1, UDA_1);
;    0874  2  Insert_Strt_Qual (EVRLA_1, UDA_1);
;    0875  2  Insert_Dev_Name (EVRLA_1, UDA_1);
;    0876  2  Insert_Tst_Strt (EVRLA_1, UDA_1);
;    0877  2  Insert_RunTime  (EVRLA_1, UDA_1);
;    0878  2  Insert_Prep_Err (EVRLA_1, UDA_1);
;    0879  2
;    0880  2  Insert_Diagnostic_Name (EVDLB, UDA_1);
;    0881  2  Insert_Set_Flgs (EVDLB, UDA_1);
;    0882  2  Insert_Set_Evnt (EVDLB, UDA_1);
;    0883  2  Insert_Strt_Qual (EVDLB, UDA_1);
;    0884  2  Insert_Dev_Name (EVDLB, UDA_1);
;    0885  2  Insert_Tst_Strt (EVDLB, UDA_1);
;    0886  2  Insert_RunTime  (EVDLB, UDA_1);
;    0887  2  Insert_Prep_Err (EVDLB, UDA_1);
;    0888  2
;    0889  2  Insert_Diagnostic_Name (EVDLC, UDA_1);
;    0890  2  Insert_Set_Flgs (EVDLC, UDA_1);
;    0891  2  Insert_Set_Evnt (EVDLC, UDA_1);
;    0892  2  Insert_Strt_Qual (EVDLC, UDA_1);
;    0893  2  Insert_Dev_Name (EVDLC, UDA_1);
;    0894  2  Insert_Tst_Strt (EVDLC, UDA_1);
```

```
;    0895  2  Insert_RunTime  (EVDLC, UDA_1);
;    0896  2  Insert_Prep_Err (EVDLC, UDA_1);
;    0897  2
;    0898  2  Insert_Diagnostic_Name (EVDLD, UDA_1);
;    0899  2  Insert_Set_Flgs (EVDLD, UDA_1);
;    0900  2  Insert_Set_Evnt (EVDLD, UDA_1);
;    0901  2  Insert_Strt_Qual (EVDLD, UDA_1);
;    0902  2  Insert_Dev_Name (EVDLD, UDA_1);
;    0903  2  Insert_Tst_Strt (EVDLD, UDA_1);
;    0904  2  Insert_RunTime  (EVDLD, UDA_1);
;    0905  2  Insert_Prep_Err (EVDLD, UDA_1);
;    0906  2
```

```
;    0907  2
;    0908  2
;    0909  2 !+
;    0910  2 ! Initialize Hardware Support Table for UDA based system with no RA60 for Drive 1    [04]
;    0911  2 ! and RA60 for Drive 0
;    0912  2 !-
;    0913  2
;    0914  2  !
;    0915  2  ! Initialize the UDA_2 Hardware Support Table to all zero pointers.
;    0916  2  !
;    0917  2
;    0918  2  INCR Diag FROM 0 TO (CRD$K_UDA_2_Last_Diagnostic - 1) DO
;    0919  2   INCR Entry FROM 0 TO (CRD$K_Last_Entry - 1) DO
;    0920  3    BEGIN
;    0921  3    CRD$AL_UDA_2_Hdwr_Sprt_Table [.Diag, .Entry] = 0;
;    0922  2    END;
;    0923  2
;    0924  2  !
;    0925  2  ! Insert pointers to the ASCIC strings into the UDA_2 Hardware Support Table
;    0926  2  !
;    0927  2
;    0928  2  Insert_Diagnostic_Name (EVRLA_0, UDA_2);
;    0929  2  Insert_Set_Flgs (EVRLA_0, UDA_2);
;    0930  2  Insert_Set_Evnt (EVRLA_0, UDA_2);
;    0931  2  Insert_Strt_Qual (EVRLA_0, UDA_2);
;    0932  2  Insert_Dev_Name (EVRLA_0, UDA_2);
;    0933  2  Insert_Tst_Strt (EVRLA_0, UDA_2);
;    0934  2  Insert_RunTime  (EVRLA_0, UDA_2);
;    0935  2  Insert_Prep_Err (EVRLA_0, UDA_2);
;    0936  2
;    0937  2  Insert_Diagnostic_Name (EVDLB, UDA_2);
;    0938  2  Insert_Set_Flgs (EVDLB, UDA_2);
;    0939  2  Insert_Set_Evnt (EVDLB, UDA_2);
;    0940  2  Insert_Strt_Qual (EVDLB, UDA_2);
;    0941  2  Insert_Dev_Name (EVDLB, UDA_2);
;    0942  2  Insert_Tst_Strt (EVDLB, UDA_2);
;    0943  2  Insert_RunTime  (EVDLB, UDA_2);
;    0944  2  Insert_Prep_Err (EVDLB, UDA_2);
;    0945  2
;    0946  2  Insert_Diagnostic_Name (EVDLC, UDA_2);
;    0947  2  Insert_Set_Flgs (EVDLC, UDA_2);
;    0948  2  Insert_Set_Evnt (EVDLC, UDA_2);
;    0949  2  Insert_Strt_Qual (EVDLC, UDA_2);
;    0950  2  Insert_Dev_Name (EVDLC, UDA_2);
;    0951  2  Insert_Tst_Strt (EVDLC, UDA_2);
;    0952  2  Insert_RunTime  (EVDLC, UDA_2);
;    0953  2  Insert_Prep_Err (EVDLC, UDA_2);
;    0954  2
;    0955  2  Insert_Diagnostic_Name (EVDLD, UDA_2);
;    0956  2  Insert_Set_Flgs (EVDLD, UDA_2);
;    0957  2  Insert_Set_Evnt (EVDLD, UDA_2);
;    0958  2  Insert_Strt_Qual (EVDLD, UDA_2);
;    0959  2  Insert_Dev_Name (EVDLD, UDA_2);
;    0960  2  Insert_Tst_Strt (EVDLD, UDA_2);
;    0961  2  Insert_RunTime  (EVDLD, UDA_2);
```

```
;    0962  2  Insert_Prep_Err (EVDLD, UDA_2);
;    0963  2
;    0964  2
```

```
;    0965  2
;    0966  2
;    0967  2
;    0968  2 !+
;    0969  2 ! Initialize Hardware Support Table for UDA based system with RA60 for Drive 1     [04]
;    0970  2 ! and RA60 for Drive 0
;    0971  2 !-
;    0972  2
;    0973  2  !
;    0974  2  ! Initialize the UDA_3 Hardware Support Table to all zero pointers.
;    0975  2  !
;    0976  2
;    0977  2  INCR Diag FROM 0 TO (CRD$K_UDA_3_Last_Diagnostic - 1) DO
;    0978  2   INCR Entry FROM 0 TO (CRD$K_Last_Entry - 1) DO
;    0979  3    BEGIN
;    0980  3    CRD$AL_UDA_3_Hdwr_Sprt_Table [.Diag, .Entry] = 0;
;    0981  2    END;
;    0982  2
;    0983  2  !
;    0984  2  ! Insert pointers to the ASCIC strings into the UDA_3 Hardware Support Table
;    0985  2  !
;    0986  2
;    0987  2  Insert_Diagnostic_Name (EVRLA_0, UDA_3);
;    0988  2  Insert_Set_Flgs (EVRLA_0, UDA_3);
;    0989  2  Insert_Set_Evnt (EVRLA_0, UDA_3);
;    0990  2  Insert_Strt_Qual (EVRLA_0, UDA_3);
;    0991  2  Insert_Dev_Name (EVRLA_0, UDA_3);
;    0992  2  Insert_Tst_Strt (EVRLA_0, UDA_3);
;    0993  2  Insert_RunTime  (EVRLA_0, UDA_3);
;    0994  2  Insert_Prep_Err (EVRLA_0, UDA_3);
;    0995  2
;    0996  2  Insert_Diagnostic_Name (EVRLA_1, UDA_3);
;    0997  2  Insert_Set_Flgs (EVRLA_1, UDA_3);
;    0998  2  Insert_Set_Evnt (EVRLA_1, UDA_3);
;    0999  2  Insert_Strt_Qual (EVRLA_1, UDA_3);
;    1000  2  Insert_Dev_Name (EVRLA_1, UDA_3);
;    1001  2  Insert_Tst_Strt (EVRLA_1, UDA_3);
;    1002  2  Insert_RunTime  (EVRLA_1, UDA_3);
;    1003  2  Insert_Prep_Err (EVRLA_1, UDA_3);
;    1004  2
;    1005  2  Insert_Diagnostic_Name (EVDLB, UDA_3);
;    1006  2  Insert_Set_Flgs (EVDLB, UDA_3);
;    1007  2  Insert_Set_Evnt (EVDLB, UDA_3);
;    1008  2  Insert_Strt_Qual (EVDLB, UDA_3);
;    1009  2  Insert_Dev_Name (EVDLB, UDA_3);
;    1010  2  Insert_Tst_Strt (EVDLB, UDA_3);
;    1011  2  Insert_RunTime  (EVDLB, UDA_3);
;    1012  2  Insert_Prep_Err (EVDLB, UDA_3);
;    1013  2
;    1014  2  Insert_Diagnostic_Name (EVDLC, UDA_3);
;    1015  2  Insert_Set_Flgs (EVDLC, UDA_3);
;    1016  2  Insert_Set_Evnt (EVDLC, UDA_3);
;    1017  2  Insert_Strt_Qual (EVDLC, UDA_3);
;    1018  2  Insert_Dev_Name (EVDLC, UDA_3);
;    1019  2  Insert_Tst_Strt (EVDLC, UDA_3);
```

```
;    1020  2  Insert_RunTime  (EVDLC, UDA_3);
;    1021  2  Insert_Prep_Err (EVDLC, UDA_3);
;    1022  2
;    1023  2  Insert_Diagnostic_Name (EVDLD, UDA_3);
;    1024  2  Insert_Set_Figs (EVDLD, UDA_3);
;    1025  2  Insert_Set_Evnt (EVDLD, UDA_3);
;    1026  2  Insert_Strt_Qual (EVDLD, UDA_3);
;    1027  2  Insert_Dev_Name (EVDLD, UDA_3);
;    1028  2  Insert_Tst_Strt (EVDLD, UDA_3);
;    1029  2  Insert_RunTime  (EVDLD, UDA_3);
;    1030  2  Insert_Prep_Err (EVDLD, UDA_3);
;    1031  2
;    1032  2  RETURN;
;    1033  1 END;


            .TITLE  CRDHDWSUP *** CRDHDWSUP Module
            .IDENT  \01-05\

            .PSECT  WORK,NOEXE,2

       00  00000 CRD$GB_BASE_SYSTEM::
            .BYTE   0               ;
            00001   .BLKB   3
        00000000  00004 CRD$GA_HDWR_SPRT_TABLE::
            .LONG   0               ;
            00008 CRD$AL_IDC_HDWR_SPRT_TABLE::
            .BLKB   160
            000A8 CRD$AL_LESI_HDWR_SPRT_TABLE::
            .BLKB   96
            00108 CRD$AL_UDA_0_HDWR_SPRT_TABLE::
            .BLKB   128
            00188 CRD$AL_UDA_1_HDWR_SPRT_TABLE::
            .BLKB   160
            00228 CRD$AL_UDA_2_HDWR_SPRT_TABLE::
            .BLKB   128
            002A8 CRD$AL_UDA_3_HDWR_SPRT_TABLE::
            .BLKB   160

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      50  55  53  57  44  48  44  52  43  09  00000 P.AAA:  .ASCII  <9>\CRDHDWSUP\          ;
        44  41  52  56  45  05  0000A P.AAB:  .ASCII  <5>\EVRAD\            ;
        4B  43  49  55  51  05  00010 P.AAC:  .ASCII  <5>\QUICK\            ;
       00  00016 P.AAD:  .ASCII  <0>              ;
     30  41  51  44  04  00017 P.AAE:  .ASCII  <4>\DQA0\            ;
       00  0001C P.AAF:  .ASCII  <0>              ;
20  20  20  20  20  20  20  20  20  20  20  20  20  12  0001D P.AAG:  .ASCII  <18>\                DQA0\         ;
          30  41  51  44  0002C                 ;
     30  33  3A  30  04  00030 P.AAH:  .ASCII  <4>\0:30\            ;
49  52  44  20  4B  53  49  44  20  3A  30  41  51  44  23  00035 P.AAI:  .ASCII  \#DQA0: DISK DRIVE TESTING INCOMPLETE\       ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  00044                 ;
            45  54  45  4C  50  4D  00053                 ;
            44  41  52  56  45  05  00059 P.AAJ:  .ASCII  <5>\EVRAD\            ;
            4B  43  49  55  51  05  0005F P.AAK:  .ASCII  <5>\QUICK\            ;
```

```
                00  00065 P.AAL:  .ASCII  <0>                ;
             31  41  51  44  04  00066 P.AAM:  .ASCII  <4>\DQA1\           ;
                00  0006B P.AAN:  .ASCII  <0>                ;
20  20  20  32  20  54  52  41  50  20  32  30  4C  52  12  0006C P.AAO:  .ASCII  <18>\RL02 PART 2   DQA1\        ;
                31  41  51  44  0007B                ;
             30  33  3A  30  04  0007F P.AAP:  .ASCII  <4>\0:30\           ;
49  52  44  20  4B  53  49  44  20  3A  31  41  51  44  23  00084 P.AAQ:  .ASCII  \#DQA1: DISK DRIVE TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  00093                ;
                45  54  45  4C  50  4D  000A2                ;
                42  4C  44  56  45  05  000A8 P.AAR:  .ASCII  <5>\EVDLB\          ;
                00  000AE P.AAS:  .ASCII  <0>                ;
                00  000AF P.AAT:  .ASCII  <0>                ;
             30  41  47  58  04  000B0 P.AAU:  .ASCII  <4>\XGA0\           ;
                00  000B5 P.AAV:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  53  32  33  46  4D  44  12  000B6 P.AAW:  .ASCII  <18>\DMF32S        XGA0\        ;
                30  41  47  58  000C5                ;
             30  30  3A  31  04  000C9 P.AAX:  .ASCII  <4>\1:00\           ;
4F  42  20  4F  42  4D  4F  43  20  3A  30  41  47  58  24  000CE P.AAY:  .ASCII  \$XGA0: COMBO BOARD TESTING INCOMPLETE\     ;
43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  41  000DD                ;
             45  54  45  4C  50  4D  4F  000EC                ;
                43  4C  44  56  45  05  000F3 P.AAZ:  .ASCII  <5>\EVDLC\          ;
                4B  43  49  55  51  05  000F9 P.ABA:  .ASCII  <5>\QUICK\          ;
                00  000FF P.ABB:  .ASCII  <0>                ;
                41  58  54  03  00100 P.ABC:  .ASCII  <3>\TXA\            ;
                00  00104 P.ABD:  .ASCII  <0>                ;
2D  30  41  58  54  20  20  20  41  32  33  46  4D  44  12  00105 P.ABE:  .ASCII  <18>\DMF32A   TXA0-TXA7\        ;
                37  41  58  54  00114                ;
             30  33  3A  30  04  00118 P.ABF:  .ASCII  <4>\0:30\           ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  58  54  23  0011D P.ABG:  .ASCII  \#TXA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  0012C                ;
                45  54  45  4C  50  4D  0013B                ;
                44  4C  44  56  45  05  00141 P.ABH:  .ASCII  <5>\EVDLD\          ;
                00  00147 P.ABI:  .ASCII  <0>                ;
                00  00148 P.ABJ:  .ASCII  <0>                ;
                41  43  4C  03  00149 P.ABK:  .ASCII  <3>\LCA\            ;
                00  0014D P.ABL:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  50  32  33  46  4D  44  12  0014E P.ABM:  .ASCII  <18>\DMF32P         LCA\        ;
                41  43  4C  20  0015D                ;
             35  31  3A  30  04  00161 P.ABN:  .ASCII  <4>\0:15\           ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  43  4C  23  00166 P.ABO:  .ASCII  \#LCA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  00175                ;
                45  54  45  4C  50  4D  00184                ;
                42  4D  52  56  45  05  0018A P.ABP:  .ASCII  <5>\EVRMB\          ;
                4B  43  49  55  51  05  00190 P.ABQ:  .ASCII  <5>\QUICK\          ;
                00  00196 P.ABR:  .ASCII  <0>                ;
             30  41  41  44  04  00197 P.ABS:  .ASCII  <4>\DAA0\           ;
 44  52  43  3A  4E  4F  49  54  43  45  53  2F  0C  0019C P.ABT:  .ASCII  <12>\/SECTION:CRD\         ;
20  20  20  32  20  54  52  41  50  20  35  32  43  52  12  001A9 P.ABU:  .ASCII  <18>\RC25 PART 2   DAA0\        ;
                30  41  41  44  001B8                ;
             30  30  3A  31  04  001BC P.ABV:  .ASCII  <4>\1:00\           ;
49  52  44  20  4B  53  49  44  20  3A  30  41  41  44  23  001C1 P.ABW:  .ASCII  \#DAA0: DISK DRIVE TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  001D0                ;
                45  54  45  4C  50  4D  001DF                ;
                42  4D  52  56  45  05  001E5 P.ABX:  .ASCII  <5>\EVRMB\          ;
                4B  43  49  55  51  05  001EB P.ABY:  .ASCII  <5>\QUICK\          ;
```

```
                00  001F1 P.ABZ:  .ASCII  <0>                ;
             31  41  41  44  04  001F2 P.ACA:  .ASCII  <4>\DAA1\          ;
 44  52  43  3A  4E  4F  49  54  43  45  53  2F  0C  001F7 P.ACB:  .ASCII  <12>\/SECTION:CRD\          ;
20  20  20  20  20  20  20  20  20  35  32  46  43  52  12  00204 P.ACC:  .ASCII  <18>\RCF25          DAA1\         ;
                31  41  41  44  00213                ;
             30  30  3A  31  04  00217 P.ACD:  .ASCII  <4>\1:00\          ;
49  52  44  20  4B  53  49  44  20  3A  31  41  41  44  23  0021C P.ACE:  .ASCII  \#DAA1: DISK DRIVE TESTING INCOMPLETE\       ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  0022B                ;
                45  54  45  4C  50  4D  0023A                ;
                41  4C  52  56  45  05  00240 P.ACF:  .ASCII  <5>\EVRLA\          ;
                4B  43  49  55  51  05  00246 P.ACG:  .ASCII  <5>\QUICK\          ;
                00  0024C P.ACH:  .ASCII  <0>                ;
             30  41  55  44  04  0024D P.ACI:  .ASCII  <4>\DUA0\          ;
32  31  44  52  43  3A  4E  4F  49  54  43  45  53  2F  0E  00252 P.ACJ:  .ASCII  <14>\/SECTION:CRD12\         ;
20  20  20  32  20  54  52  41  50  20  20  20  20  20  12  00261 P.ACK:  .ASCII  <18>\     PART 2   DUA0\         ;
                30  41  55  44  00270                ;
             35  34  3A  32  04  00274 P.ACL:  .ASCII  <4>\2:45\          ;
49  52  44  20  4B  53  49  44  20  3A  30  41  55  44  23  00279 P.ACM:  .ASCII  \#DUA0: DISK DRIVE TESTING INCOMPLETE\       ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  00288                ;
                45  54  45  4C  50  4D  00297                ;
                42  4C  44  56  45  05  0029D P.ACN:  .ASCII  <5>\EVDLB\          ;
                00  002A3 P.ACO:  .ASCII  <0>                ;
                00  002A4 P.ACP:  .ASCII  <0>                ;
             30  41  47  58  04  002A5 P.ACQ:  .ASCII  <4>\XGA0\          ;
                00  002AA P.ACR:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  53  32  33  46  4D  44  12  002AB P.ACS:  .ASCII  <18>\DMF32S          XGA0\         ;
                30  41  47  58  002BA                ;
             30  30  3A  31  04  002BE P.ACT:  .ASCII  <4>\1:00\          ;
4F  42  20  4F  42  4D  4F  43  20  3A  30  41  47  58  24  002C3 P.ACU:  .ASCII  \$XGA0: COMBO BOARD TESTING INCOMPLETE\      ;
43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  41  002D2                ;
             45  54  45  4C  50  4D  4F  002E1                ;
                43  4C  44  56  45  05  002E8 P.ACV:  .ASCII  <5>\EVDLC\          ;
                4B  43  49  55  51  05  002EE P.ACW:  .ASCII  <5>\QUICK\          ;
                00  002F4 P.ACX:  .ASCII  <0>                ;
                41  58  54  03  002F5 P.ACY:  .ASCII  <3>\TXA\           ;
                00  002F9 P.ACZ:  .ASCII  <0>                ;
2D  30  41  58  54  20  20  20  41  32  33  46  4D  44  12  002FA P.ADA:  .ASCII  <18>\DMF32A   TXA0-TXA7\         ;
                37  41  58  54  00309                ;
             30  33  3A  30  04  0030D P.ADB:  .ASCII  <4>\0:30\          ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  58  54  23  00312 P.ADC:  .ASCII  \#TXA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  00321                ;
                45  54  45  4C  50  4D  00330                ;
                44  4C  44  56  45  05  00336 P.ADD:  .ASCII  <5>\EVDLD\          ;
                00  0033C P.ADE:  .ASCII  <0>                ;
                00  0033D P.ADF:  .ASCII  <0>                ;
                41  43  4C  03  0033E P.ADG:  .ASCII  <3>\LCA\           ;
                00  00342 P.ADH:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  50  32  33  46  4D  44  12  00343 P.ADI:  .ASCII  <18>\DMF32P          LCA\         ;
                41  43  4C  20  00352                ;
             35  31  3A  30  04  00356 P.ADJ:  .ASCII  <4>\0:15\          ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  43  4C  23  0035B P.ADK:  .ASCII  \#LCA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  0036A                ;
                45  54  45  4C  50  4D  00379                ;
                41  4C  52  56  45  05  0037F P.ADL:  .ASCII  <5>\EVRLA\          ;
                4B  43  49  55  51  05  00385 P.ADM:  .ASCII  <5>\QUICK\          ;
```

```
      00  0038B P.ADN:  .ASCII  <0>                ;
    30  41  55  44  04  0038C P.ADO:  .ASCII  <4>\DUA0\              ;
32  31  44  52  43  3A  4E  4F  49  54  43  45  53  2F  0E  00391 P.ADP:  .ASCII  <14>\/SECTION:CRD12\          ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  12  003A0 P.ADQ:  .ASCII  <18>\                DUA0\          ;
        30  41  55  44  003AF                   ;
    35  34  3A  32  04  003B3 P.ADR:  .ASCII  <4>\2:45\             ;
49  52  44  20  4B  53  49  44  20  3A  30  41  55  44  23  003B8 P.ADS:  .ASCII  \#DUA0: DISK DRIVE TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  003C7                ;
        45  54  45  4C  50  4D  003D6                ;
        41  4C  52  56  45  05  003DC P.ADT:  .ASCII  <5>\EVRLA\          ;
        4B  43  49  55  51  05  003E2 P.ADU:  .ASCII  <5>\QUICK\          ;
      00  003E8 P.ADV:  .ASCII  <0>                ;
    31  41  4A  44  04  003E9 P.ADW:  .ASCII  <4>\DJA1\              ;
32  31  44  52  43  3A  4E  4F  49  54  43  45  53  2F  0E  003EE P.ADX:  .ASCII  <14>\/SECTION:CRD12\          ;
20  20  20  32  20  54  52  41  50  20  30  36  41  52  12  003FD P.ADY:  .ASCII  <18>\RA60 PART 2   DJA1\          ;
        31  41  4A  44  0040C                   ;
    30  30  3A  31  04  00410 P.ADZ:  .ASCII  <4>\1:00\              ;
49  52  44  20  4B  53  49  44  20  3A  31  41  4A  44  23  00415 P.AEA:  .ASCII  \#DJA1: DISK DRIVE TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  00424                ;
        45  54  45  4C  50  4D  00433                ;
        42  4C  44  56  45  05  00439 P.AEB:  .ASCII  <5>\EVDLB\          ;
      00  0043F P.AEC:  .ASCII  <0>                ;
      00  00440 P.AED:  .ASCII  <0>                ;
    30  41  47  58  04  00441 P.AEE:  .ASCII  <4>\XGA0\              ;
      00  00446 P.AEF:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  53  32  33  46  4D  44  12  00447 P.AEG:  .ASCII  <18>\DMF32S        XGA0\          ;
        30  41  47  58  00456                   ;
    30  30  3A  31  04  0045A P.AEH:  .ASCII  <4>\1:00\              ;
4F  42  20  4F  42  4D  4F  43  20  3A  30  41  47  58  24  0045F P.AEI:  .ASCII  \$XGA0: COMBO BOARD TESTING INCOMPLETE\      ;
43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  41  0046E                ;
    45  54  45  4C  50  4D  4F  0047D                ;
        43  4C  44  56  45  05  00484 P.AEJ:  .ASCII  <5>\EVDLC\          ;
        4B  43  49  55  51  05  0048A P.AEK:  .ASCII  <5>\QUICK\          ;
      00  00490 P.AEL:  .ASCII  <0>                ;
        41  58  54  03  00491 P.AEM:  .ASCII  <3>\TXA\            ;
      00  00495 P.AEN:  .ASCII  <0>                ;
2D  30  41  58  54  20  20  20  41  32  33  46  4D  44  12  00496 P.AEO:  .ASCII  <18>\DMF32A   TXA0-TXA7\          ;
        37  41  58  54  004A5                   ;
    30  33  3A  30  04  004A9 P.AEP:  .ASCII  <4>\0:30\              ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  58  54  23  004AE P.AEQ:  .ASCII  \#TXA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  004BD                ;
        45  54  45  4C  50  4D  004CC                ;
        44  4C  44  56  45  05  004D2 P.AER:  .ASCII  <5>\EVDLD\          ;
      00  004D8 P.AES:  .ASCII  <0>                ;
      00  004D9 P.AET:  .ASCII  <0>                ;
        41  43  4C  03  004DA P.AEU:  .ASCII  <3>\LCA\            ;
      00  004DE P.AEV:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  50  32  33  46  4D  44  12  004DF P.AEW:  .ASCII  <18>\DMF32P          LCA\          ;
        41  43  4C  20  004EE                   ;
    35  31  3A  30  04  004F2 P.AEX:  .ASCII  <4>\0:15\              ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  43  4C  23  004F7 P.AEY:  .ASCII  \#LCA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  00506                ;
        45  54  45  4C  50  4D  00515                ;
        41  4C  52  56  45  05  0051B P.AEZ:  .ASCII  <5>\EVRLA\          ;
        4B  43  49  55  51  05  00521 P.AFA:  .ASCII  <5>\QUICK\          ;
```

```
                00  00527 P.AFB:  .ASCII  <0>                ;
             30  41  4A  44  04  00528 P.AFC:  .ASCII  <4>\DJA0\             ;
32  31  44  52  43  3A  4E  4F  49  54  43  45  53  2F  0E  0052D P.AFD:  .ASCII  <14>\/SECTION:CRD12\          ;
20  20  20  32  20  54  52  41  50  20  30  36  41  52  12  0053C P.AFE:  .ASCII  <18>\RA60 PART 2   DJA0\        ;
                30  41  4A  44  0054B              ;
             30  30  3A  31  04  0054F P.AFF:  .ASCII  <4>\1:00\             ;
49  52  44  20  4B  53  49  44  20  3A  30  41  4A  44  23  00554 P.AFG:  .ASCII  \#DJA0: DISK DRIVE TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  00563                  ;
                45  54  45  4C  50  4D  00572                ;
                42  4C  44  56  45  05  00578 P.AFH:  .ASCII  <5>\EVDLB\          ;
                00  0057E P.AFI:  .ASCII  <0>                ;
                00  0057F P.AFJ:  .ASCII  <0>                ;
             30  41  47  58  04  00580 P.AFK:  .ASCII  <4>\XGA0\             ;
                00  00585 P.AFL:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  53  32  33  46  4D  44  12  00586 P.AFM:  .ASCII  <18>\DMF32S        XGA0\        ;
                30  41  47  58  00595              ;
             30  30  3A  31  04  00599 P.AFN:  .ASCII  <4>\1:00\             ;
4F  42  20  4F  42  4D  4F  43  20  3A  30  41  47  58  24  0059E P.AFO:  .ASCII  \$XGA0: COMBO BOARD TESTING INCOMPLETE\     ;
43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  41  005AD                  ;
    45  54  45  4C  50  4D  4F  005BC                ;
                43  4C  44  56  45  05  005C3 P.AFP:  .ASCII  <5>\EVDLC\          ;
                4B  43  49  55  51  05  005C9 P.AFQ:  .ASCII  <5>\QUICK\          ;
                00  005CF P.AFR:  .ASCII  <0>                ;
                41  58  54  03  005D0 P.AFS:  .ASCII  <3>\TXA\           ;
                00  005D4 P.AFT:  .ASCII  <0>                ;
2D  30  41  58  54  20  20  20  41  32  33  46  4D  44  12  005D5 P.AFU:  .ASCII  <18>\DMF32A    TXA0-TXA7\        ;
                37  41  58  54  005E4              ;
             30  33  3A  30  04  005E8 P.AFV:  .ASCII  <4>\0:30\             ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  58  54  23  005ED P.AFW:  .ASCII  \#TXA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  005FC                  ;
                45  54  45  4C  50  4D  0060B                ;
                44  4C  44  56  45  05  00611 P.AFX:  .ASCII  <5>\EVDLD\          ;
                00  00617 P.AFY:  .ASCII  <0>                ;
                00  00618 P.AFZ:  .ASCII  <0>                ;
                41  43  4C  03  00619 P.AGA:  .ASCII  <3>\LCA\           ;
                00  0061D P.AGB:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  50  32  33  46  4D  44  12  0061E P.AGC:  .ASCII  <18>\DMF32P         LCA\        ;
                41  43  4C  20  0062D              ;
             35  31  3A  30  04  00631 P.AGD:  .ASCII  <4>\0:15\             ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  43  4C  23  00636 P.AGE:  .ASCII  \#LCA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  00645                  ;
                45  54  45  4C  50  4D  00654                ;
                41  4C  52  56  45  05  0065A P.AGF:  .ASCII  <5>\EVRLA\          ;
                4B  43  49  55  51  05  00660 P.AGG:  .ASCII  <5>\QUICK\          ;
                00  00666 P.AGH:  .ASCII  <0>                ;
             30  41  4A  44  04  00667 P.AGI:  .ASCII  <4>\DJA0\             ;
32  31  44  52  43  3A  4E  4F  49  54  43  45  53  2F  0E  0066C P.AGJ:  .ASCII  <14>\/SECTION:CRD12\          ;
20  20  20  20  20  20  20  20  20  20  20  20  20  20  12  0067B P.AGK:  .ASCII  <18>\                DJA0\        ;
                30  41  4A  44  0068A              ;
             30  30  3A  31  04  0068E P.AGL:  .ASCII  <4>\1:00\             ;
49  52  44  20  4B  53  49  44  20  3A  30  41  4A  44  23  00693 P.AGM:  .ASCII  \#DJA0: DISK DRIVE TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  006A2                  ;
                45  54  45  4C  50  4D  006B1                ;
                41  4C  52  56  45  05  006B7 P.AGN:  .ASCII  <5>\EVRLA\          ;
                4B  43  49  55  51  05  006BD P.AGO:  .ASCII  <5>\QUICK\          ;
```

```
      00  006C3 P.AGP:  .ASCII  <0>                ;
    31  41  4A  44  04  006C4 P.AGQ:  .ASCII  <4>\DJA1\              ;
32  31  44  52  43  3A  4E  4F  49  54  43  45  53  2F  0E  006C9 P.AGR:  .ASCII  <14>\/SECTION:CRD12\          ;
20  20  20  32  20  54  52  41  50  20  30  36  41  52  12  006D8 P.AGS:  .ASCII  <18>\RA60 PART 2   DJA1\         ;
        31  41  4A  44  006E7                   ;
    30  30  3A  31  04  006EB P.AGT:  .ASCII  <4>\1:00\              ;
49  52  44  20  4B  53  49  44  20  3A  31  41  4A  44  23  006F0 P.AGU:  .ASCII  \#DJA1: DISK DRIVE TESTING INCOMPLETE\       ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  45  56  006FF                   ;
        45  54  45  4C  50  4D  0070E                 ;
        42  4C  44  56  45  05  00714 P.AGV:  .ASCII  <5>\EVDLB\          ;
      00  0071A P.AGW:  .ASCII  <0>                ;
      00  0071B P.AGX:  .ASCII  <0>                ;
    30  41  47  58  04  0071C P.AGY:  .ASCII  <4>\XGA0\              ;
      00  00721 P.AGZ:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  53  32  33  46  4D  44  12  00722 P.AHA:  .ASCII  <18>\DMF32S        XGA0\         ;
        30  41  47  58  00731                   ;
    30  30  3A  31  04  00735 P.AHB:  .ASCII  <4>\1:00\              ;
4F  42  20  4F  42  4D  4F  43  20  3A  30  41  47  58  24  0073A P.AHC:  .ASCII  \$XGA0: COMBO BOARD TESTING INCOMPLETE\      ;
43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  41  00749                   ;
    45  54  45  4C  50  4D  4F  00758                ;
        43  4C  44  56  45  05  0075F P.AHD:  .ASCII  <5>\EVDLC\          ;
        4B  43  49  55  51  05  00765 P.AHE:  .ASCII  <5>\QUICK\          ;
      00  0076B P.AHF:  .ASCII  <0>                ;
      41  58  54  03  0076C P.AHG:  .ASCII  <3>\TXA\            ;
      00  00770 P.AHH:  .ASCII  <0>                ;
2D  30  41  58  54  20  20  20  41  32  33  46  4D  44  12  00771 P.AHI:  .ASCII  <18>\DMF32A   TXA0-TXA7\         ;
        37  41  58  54  00780                   ;
    30  33  3A  30  04  00784 P.AHJ:  .ASCII  <4>\0:30\              ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  58  54  23  00789 P.AHK:  .ASCII  \#TXA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  00798                   ;
        45  54  45  4C  50  4D  007A7                 ;
        44  4C  44  56  45  05  007AD P.AHL:  .ASCII  <5>\EVDLD\          ;
      00  007B3 P.AHM:  .ASCII  <0>                ;
      00  007B4 P.AHN:  .ASCII  <0>                ;
      41  43  4C  03  007B5 P.AHO:  .ASCII  <3>\LCA\            ;
      00  007B9 P.AHP:  .ASCII  <0>                ;
20  20  20  20  20  20  20  20  50  32  33  46  4D  44  12  007BA P.AHQ:  .ASCII  <18>\DMF32P         LCA\         ;
        41  43  4C  20  007C9                   ;
    35  31  3A  30  04  007CD P.AHR:  .ASCII  <4>\0:15\              ;
41  4F  42  20  4F  42  4D  4F  43  20  3A  41  43  4C  23  007D2 P.AHS:  .ASCII  \#LCA: COMBO BOARD TESTING INCOMPLETE\      ;
4F  43  4E  49  20  47  4E  49  54  53  45  54  20  44  52  007E1                   ;
        45  54  45  4C  50  4D  007F0                 ;
        20  44  41  4F  4C  05  007F6 P.AHT:  .ASCII  <5>\LOAD \          ;
  20  53  47  41  4C  46  20  54  45  53  0A  007FC P.AHU:  .ASCII  <10>\SET FLAGS \          ;
47  41  4C  46  20  54  4E  45  56  45  20  54  45  53  10  00807 P.AHV:  .ASCII  <16>\SET EVENT FLAGS \        ;
        20  53  00816                   ;
      20  54  43  45  4C  45  53  07  00818 P.AHW:  .ASCII  <7>\SELECT \           ;
    20  54  52  41  54  53  06  00820 P.AHX:  .ASCII  <6>\START \           ;

        $MODULE=        P.AAA
        CRD$GT_IDC_EVRAD_0= P.AAB
        CRD$GT_IDC_EVRAD_0_SET_FLGS=
              P.AAC
        CRD$GT_IDC_EVRAD_0_SET_EVNT-
              P.AAD
```

```
          CRD$GT_IDC_EVRAD_0_DEV_NAME=
                P.AAE
          CRD$GT_IDC_EVRAD_0_STRT_QUAL=
                P.AAF
          CRD$GT_IDC_EVRAD_0_TST_STRT=
                P.AAG
          CRD$GT_IDC_EVRAD_0_RUNTIME=
                P.AAH
          CRD$GT_IDC_EVRAD_0_PREP_ERR=
                P.AAI
          CRD$GT_IDC_EVRAD_1= P.AAJ
          CRD$GT_IDC_EVRAD_1_SET_FLGS=
                P.AAK
          CRD$GT_IDC_EVRAD_1_SET_EVNT=
                P.AAL
          CRD$GT_IDC_EVRAD_1_DEV_NAME=
                P.AAM
          CRD$GT_IDC_EVRAD_1_STRT_QUAL=
                P.AAN
          CRD$GT_IDC_EVRAD_1_TST_STRT=
                P.AAO
          CRD$GT_IDC_EVRAD_1_RUNTIME=
                P.AAP
          CRD$GT_IDC_EVRAD_1_PREP_ERR=
                P.AAQ
          CRD$GT_IDC_EVDLB=   P.AAR
          CRD$GT_IDC_EVDLB_SET_FLGS-
                P.AAS
          CRD$GT_IDC_EVDLB_SET_EVNT=
                P.AAT
          CRD$GT_IDC_EVDLB_DEV_NAME=
                P.AAU
          CRD$GT_IDC_EVDLB_STRT_QUAL=
                P.AAV
          CRD$GT_IDC_EVDLB_TST_STRT=
                P.AAW
          CRD$GT_IDC_EVDLB_RUNTIME=
                P.AAX
          CRD$GT_IDC_EVDLB_PREP_ERR=
                P.AAY
          CRD$GT_IDC_EVDLC=   P.AAZ
          CRD$GT_IDC_EVDLC_SET_FLGS=
                P.ABA
          CRD$GT_IDC_EVDLC_SET_EVNT=
                P.ABB
          CRD$GT_IDC_EVDLC_DEV_NAME=
                P.ABC
          CRD$GT_IDC_EVDLC_STRT_QUAL=
                P.ABD
          CRD$GT_IDC_EVDLC_TST_STRT=
                P.ABE
          CRD$GT_IDC_EVDLC_RUNTIME=
                P.ABF
          CRD$GT_IDC_EVDLC_PREP_ERR=
                P.ABG
```

```
          CRD$GT_IDC_EVDLD=   P.ABH
          CRD$GT_IDC_EVDLD_SET_FLGS=
                P.ABI
          CRD$GT_IDC_EVDLD_SET_EVNT=
                P.ABJ
          CRD$GT_IDC_EVDLD_DEV_NAME=
                P.ABK
          CRD$GT_IDC_EVDLD_STRT_QUAL=
                P.ABL
          CRD$GT_IDC_EVDLD_TST_STRT=
                P.ABM
          CRD$GT_IDC_EVDLD_RUNTIME=
                P.ABN
          CRD$GT_IDC_EVDLD_PREP_ERR=
                P.ABO
          CRD$GT_LESI_EVRMB_0=P.ABP
          CRD$GT_LESI_EVRMB_0_SET_FLGS=
                P.ABQ
          CRD$GT_LESI_EVRMB_0_SET_EVNT=
                P.ABR
          CRD$GT_LESI_EVRMB_0_DEV_NAME=
                P.ABS
          CRD$GT_LESI_EVRMB_0_STRT_QUAL=
                P.ABT
          CRD$GT_LESI_EVRMB_0_TST_STRT=
                P.ABU
          CRD$GT_LESI_EVRMB_0_RUNTIME=
                P.ABV
          CRD$GT_LESI_EVRMB_0_PREP_ERR=
                P.ABW
          CRD$GT_LESI_EVRMB_1=P.ABX
          CRD$GT_LESI_EVRMB_1_SET_FLGS=
                P.ABY
          CRD$GT_LESI_EVRMB_1_SET_EVNT=
                P.ABZ
          CRD$GT_LESI_EVRMB_1 DEV_NAME=
                P.ACA
          CRD$GT_LESI_EVRMB_1 STRT_QUAL=
                P.ACB
          CRD$GT_LESI_EVRMB_1 TST_STRT=
                P.ACC
          CRD$GT_LESI_EVRMB_1_RUNTIME=
                P.ACD
          CRD$GT_LESI_EVRMB_1_PREP_ERR=
                P.ACE
          CRD$GT_UDA_0_EVRLA_0=
                P.ACF
          CRD$GT_UDA_0_EVRLA_0_SET_FLGS=
                P.ACG
          CRD$GT_UDA_0_EVRLA_0_SET_EVNT=
                P.ACH
          CRD$GT_UDA_0_EVRLA_0_DEV_NAME=
                P.ACI
          CRD$GT_UDA 0_EVRLA_0_STRT_QUAL=
                P.ACJ
```

```
          CRD$GT_UDA_0_EVRLA_0_TST_STRT=
                P.ACK
          CRD$GT_UDA_0_EVRLA_0_RUNTIME=
                P.ACL
          CRD$GT_UDA_0_EVRLA_0_PREP_ERR=
                P.ACM
          CRD$GT_UDA_0_EVDLB= P.ACN
          CRD$GT_UDA_0_EVDLB_SET_FLGS=
                P.ACO
          CRD$GT_UDA_0_EVDLB_SET_EVNT=
                P.ACP
          CRD$GT_UDA_0_EVDLB_DEV_NAME=
                P.ACQ
          CRD$GT_UDA_0_EVDLB_STRT_QUAL=
                P.ACR
          CRD$GT_UDA_0_EVDLB_TST_STRT=
                P.ACS
          CRD$GT_UDA_0_EVDLB_RUNTIME=
                P.ACT
          CRD$GT_UDA_0_EVDLB_PREP_ERR=
                P.ACU
          CRD$GT_UDA_0_EVDLC= P.ACV
          CRD$GT_UDA_0_EVDLC_SET_FLGS=
                P.ACW
          CRD$GT_UDA_0_EVDLC_SET_EVNT=
                P.ACX
          CRD$GT_UDA_0_EVDLC_DEV NAME=
                P.ACY
          CRD$GT_UDA_0_EVDLC_STRT QUAL-
                P.ACZ
          CRD$GT_UDA_0_EVDLC_TST_STRT-
                P.ADA
          CRD$GT_UDA_0_EVDLC RUNTIME=
                P.ADB
          CRD$GT_UDA_0_EVDLC_PREP ERR-
                P.ADC
          CRD$GT_UDA_0_EVDLD= P.ADD
          CRD$GT_UDA_0_EVDLD SET_FLGS=
                P.ADE
          CRD$GT_UDA_0_EVDLD SET EVNT-
                P.ADF
          CRD$GT_UDA_0_EVDLD_DEV_NAME-
                P.ADG
          CRD$GT_UDA_0_EVDLD_STRT_QUAL=
                P.ADH
          CRD$GT_UDA_0_EVDLD_TST_STRT=
                P.ADI
          CRD$GT_UDA_0_EVDLD_RUNTIME=
                P.ADJ
          CRD$GT_UDA_0_EVDLD_PREP_ERR=
                P.ADK
          CRD$GT_UDA_1_EVRLA_0=
                P.ADL
          CRD$GT_UDA_1_EVRLA_0_SET_FLGS=
                P.ADM
```

```
          CRD$GT_UDA_1_EVRLA_0_SET_EVNT=
                P.ADN
          CRD$GT_UDA_1_EVRLA_0_DEV_NAME=
                P.ADO
          CRD$GT_UDA_1_EVRLA_0_STRT_QUAL=
                P.ADP
          CRD$GT_UDA_1_EVRLA_0_TST_STRT=
                P.ADQ
          CRD$GT_UDA_1_EVRLA_0_RUNTIME=
                P.ADR
          CRD$GT_UDA_1_EVRLA_0_PREP_ERR=
                P.ADS
          CRD$GT_UDA_1_EVRLA_1=
                P.ADT
          CRD$GT_UDA_1_EVRLA_1_SET_FLGS=
                P.ADU
          CRD$GT_UDA_1_EVRLA_1_SET_EVNT=
                P.ADV
          CRD$GT_UDA_1_EVRLA_1_DEV_NAME=
                P.ADW
          CRD$GT_UDA_1_EVRLA_1_STRT_QUAL=
                P.ADX
          CRD$GT_UDA_1_EVRLA_1_TST_STRT=
                P.ADY
          CRD$GT_UDA_1_EVRLA_1_RUNTIME=
                P.ADZ
          CRD$GT_UDA_1_EVRLA_1_PREP_ERR=
                P.AEA
          CRD$GT_UDA_1_EVDLB= P.AEB
          CRD$GT_UDA_1_EVDLB_SET_FLGS-
                P.AEC
          CRD$GT_UDA_1_EVDLB_SET_EVNT=
                P.AED
          CRD$GT_UDA_1_EVDLB_DEV NAME-
                P.AEE
          CRD$GT_UDA_1 EVDLB_STRT QUAL-
                P.AEF
          CRD$GT_UDA_1_EVDLB_TST_STRT-
                P.AEG
          CRD$GT_UDA_1_EVDLB_RUNTIME=
                P.AEH
          CRD$GT_UDA_1_EVDLB_PREP_ERR=
                P.AEI
          CRD$GT_UDA_1_EVDLC= P.AEJ
          CRD$GT_UDA_1 EVDLC SET FLGS=
                P.AEK
          CRD$GT_UDA_1_EVDLC SET_EVNT=
                P.AEL
          CRD$GT_UDA 1_EVDLC_DEV_NAME=
                P.AEM
          CRD$GT_UDA_1_EVDLC_STRT_QUAL=
                P.AEN
          CRD$GT_UDA_1_EVDLC_TST_STRT=
                P.AEO
          CRD$GT_UDA 1_EVDLC RUNTIME-
```

```
                P.AEP
            CRD$GT_UDA_1_EVDLC_PREP_ERR=
                P.AEQ
            CRD$GT_UDA_1_EVDLD= P.AER
            CRD$GT_UDA_1_EVDLD_SET_FLGS=
                P.AES
            CRD$GT_UDA_1_EVDLD_SET_EVNT=
                P.AET
            CRD$GT_UDA_1_EVDLD_DEV_NAME=
                P.AEU
            CRD$GT_UDA_1_EVDLD_STRT_QUAL=
                P.AEV
            CRD$GT_UDA_1_EVDLD_TST_STRT=
                P.AEW
            CRD$GT_UDA_1_EVDLD_RUNTIME=
                P.AEX
            CRD$GT_UDA_1_EVDLD_PREP_ERR=
                P.AEY
            CRD$GT_UDA_2_EVRLA_0-
                P.AEZ
            CRD$GT_UDA_2_EVRLA_0_SET_FLGS=
                P.AFA
            CRD$GT_UDA_2_EVRLA_0_SET_EVNT=
                P.AFB
            CRD$GT_UDA_2_EVRLA_0_DEV_NAME=
                P.AFC
            CRD$GT_UDA_2_EVRLA_0_STRT_QUAL=
                P.AFD
            CRD$GT_UDA_2_EVRLA_0_TST_STRT=
                P.AFE
            CRD$GT_UDA_2_EVRLA_0_RUNTIME=
                P.AFF
            CRD$GT_UDA_2_EVRLA_0_PREP_ERR-
                P.AFG
            CRD$GT_UDA_2_EVDLB= P.AFH
            CRD$GT_UDA_2_EVDLB SET_FLGS=
                P.AFI
            CRD$GT_UDA_2_EVDLB_SET_EVNT-
                P.AFJ
            CRD$GT_UDA_2_EVDLB_DEV_NAME=
                P.AFK
            CRD$GT_UDA_2_EVDLB_STRT_QUAL=
                P.AFL
            CRD$GT_UDA_2_EVDLB_TST_STRT=
                P.AFM
            CRD$GT_UDA_2_EVDLB RUNTIME=
                P.AFN
            CRD$GT_UDA_2_EVDLB PREP ERR=
                P.AFO
            CRD$GT_UDA_2_EVDLC- P.AFP
            CRD$GT_UDA_2_EVDLC_SET_FLGS-
                P.AFQ
            CRD$GT_UDA_2_EVDLC_SET_EVNT-
                P.AFR
            CRD$GT_UDA_2_EVDLC DEV_NAME-
```

```
                P.AFS
        CRD$GT_UDA_2_EVDLC_STRT_QUAL=
                P.AFT
        CRD$GT_UDA_2_EVDLC_TST_STRT=
                P.AFU
        CRD$GT_UDA_2_EVDLC_RUNTIME=
                P.AFV
        CRD$GT_UDA_2_EVDLC_PREP_ERR=
                P.AFW
        CRD$GT_UDA_2_EVDLD= P.AFX
        CRD$GT_UDA_2_EVDLD_SET_FLGS=
                P.AFY
        CRD$GT_UDA_2_EVDLD_SET_EVNT=
                P.AFZ
        CRD$GT_UDA_2_EVDLD_DEV_NAME-
                P.AGA
        CRD$GT_UDA_2_EVDLD_STRT_QUAL=
                P.AGB
        CRD$GT_UDA_2_EVDLD_TST_STRT=
                P.AGC
        CRD$GT_UDA_2_EVDLD_RUNTIME=
                P.AGD
        CRD$GT_UDA_2_EVDLD_PREP_ERR=
                P.AGE
        CRD$GT_UDA_3_EVRLA_0=
                P.AGF
        CRD$GT_UDA_3_EVRLA_0_SET_FLGS=
                P.AGG
        CRD$GT_UDA_3_EVRLA_0_SET_EVNT=
                P.AGH
        CRD$GT_UDA_3_EVRLA_0_DEV_NAME=
                P.AGI
        CRD$GT_UDA_3_EVRLA_0_STRT_QUAL=
                P.AGJ
        CRD$GT_UDA_3_EVRLA_0_TST_STRT=
                P.AGK
        CRD$GT_UDA_3_EVRLA_0_RUNTIME=
                P.AGL
        CRD$GT_UDA_3_EVRLA_0_PREP_ERR=
                P.AGM
        CRD$GT_UDA_3_EVRLA_1=
                P.AGN
        CRD$GT_UDA_3_EVRLA_1_SET_FLGS=
                P.AGO
        CRD$GT_UDA_3_EVRLA_1_SET_EVNT=
                P.AGP
        CRD$GT_UDA_3_EVRLA_1_DEV_NAME=
                P.AGQ
        CRD$GT_UDA_3_EVRLA_1_STRT_QUAL=
                P.AGR
        CRD$GT_UDA_3_EVRLA_1_TST_STRT=
                P.AGS
        CRD$GT_UDA_3_EVRLA_1_RUNTIME=
                P.AGT
        CRD$GT_UDA_3_EVRLA_1_PREP_ERR=
```

```
                P.AGU
        CRD$GT_UDA_3_EVDLB= P.AGV
        CRD$GT_UDA_3_EVDLB_SET_FLGS=
                P.AGW
        CRD$GT_UDA_3_EVDLB_SET_EVNT=
                P.AGX
        CRD$GT_UDA_3_EVDLB_DEV_NAME=
                P.AGY
        CRD$GT_UDA_3_EVDLB_STRT_QUAL=
                P.AGZ
        CRD$GT_UDA_3_EVDLB_TST_STRT=
                P.AHA
        CRD$GT_UDA_3_EVDLB_RUNTIME=
                P.AHB
        CRD$GT_UDA_3_EVDLB_PREP_ERR=
                P.AHC
        CRD$GT_UDA_3_EVDLC= P.AHD
        CRD$GT_UDA_3_EVDLC_SET_FLGS=
                P.AHE
        CRD$GT_UDA_3_EVDLC_SET_EVNT=
                P.AHF
        CRD$GT_UDA_3_EVDLC_DEV_NAME=
                P.AHG
        CRD$GT_UDA_3_EVDLC_STRT_QUAL-
                P.AHH
        CRD$GT_UDA_3_EVDLC_TST_STRT=
                P.AHI
        CRD$GT_UDA_3_EVDLC_RUNTIME=
                P.AHJ
        CRD$GT_UDA_3_EVDLC_PREP_ERR=
                P.AHK
        CRD$GT_UDA_3_EVDLD= P.AHL
        CRD$GT_UDA_3_EVDLD_SET_FLGS=
                P.AHM
        CRD$GT_UDA_3_EVDLD_SET_EVNT=
                P.AHN
        CRD$GT_UDA_3_EVDLD_DEV_NAME=
                P.AHO
        CRD$GT_UDA_3_EVDLD_STRT_QUAL=
                P.AHP
        CRD$GT_UDA_3_EVDLD TST_STRT=
                P.AHQ
        CRD$GT_UDA_3_EVDLD RUNTIME=
                P.AHR
        CRD$GT_UDA_3_EVDLD_PREP_ERR=
                P.AHS
        CRD$GT_DS_LOAD_COM--P.AHT
        CRD$GT_DS_SET_FLAGS_COM==
                P.AHU
        CRD$GT_DS_SET_EVENT_COM-=
                P.AHV
        CRD$GT_DS_SELECT_COM--
                P.AHW
        CRD$GT_DS_START COM=-
                P.AHX
```

```
                .EXTRN  CRD$BUILD_SUPPORT_TABLE
                .EXTRN  CRD$PRINT_HARDWARE_SUPPORT

                .PSECT  CODE,NOWRT,  SHR,  PIC,2

                  0004 00000     .ENTRY  CRD$BUILD_SUPPORT_TABLE, Save R2        ; 0563
                52  D4 00002     CLRL    DIAG                    ; 0678
                51  D4 00004 1$:   CLRL    ENTRY                   ; 0679
                50    6241  DE 00006 2$:   MOVAL   (DIAG)[ENTRY], R0          ; 0681
                05      52  D1 0000A     CMPL    DIAG, #5                ;
                05  18 0000D     BGEQ    3$                      ;
                08      51  D1 0000F     CMPL    ENTRY, #8               ;
                05  19 00012     BLSS    4$                      ;
                50      01  CE 00014 3$:   MNEGL   #1, R0                ;
                08  11 00017     BRB     5$                      ;
                50 00000000'EF40  9E 00019 4$:   MOVAB   CRD$AL_IDC_HDWR_SPRT_TABLE[R0], R0        ;
                60  D4 00021 5$:   CLRL    (R0)                    ;
                DF        51        07  F3 00023    AOBLEQ  #7, ENTRY, 2$           ; 0679
  FFD3          52        20 00000080   8F  F1 00027    ACBL    #128, #32, DIAG, 1$         ;
      00000000'  EF 00000000'  EF  9E 00031    MOVAB   CRD$GT_IDC_EVRAD_0, -           ; 0688
           CRD$AL_IDC_HDWR_SPRT_TABLE          ;
      00000000'  EF 00000000'  EF  9E 0003C    MOVAB   CRD$GT_IDC_EVRAD_0_SET_FLGS, -      ; 0689
           CRD$AL_IDC_HDWR_SPRT_TABLE+4        ;
      00000000'  EF 00000000'  EF  9E 00047    MOVAB   CRD$GT_IDC_EVRAD_0_SET_EVNT, -      ; 0690
           CRD$AL_IDC_HDWR_SPRT_TABLE+8        ;
      00000000'  EF 00000000'  EF  9E 00052    MOVAB   CRD$GT_IDC_EVRAD_0_STRT_QUAL, -     ; 0691
           CRD$AL_IDC_HDWR_SPRT_TABLE+16       ;
      00000000'  EF 00000000'  EF  9E 0005D    MOVAB   CRD$GT_IDC_EVRAD_0_DEV_NAME, -      ; 0692
           CRD$AL_IDC_HDWR_SPRT_TABLE+12       ;
      00000000'  EF 00000000'  EF  9E 00068    MOVAB   CRD$GT_IDC_EVRAD_0_TST_STRT, -      ; 0693
           CRD$AL_IDC_HDWR_SPRT_TABLE+20       ;
      00000000'  EF 00000000'  EF  9E 00073    MOVAB   CRD$GT_IDC_EVRAD_0_RUNTIME, -       ; 0694
           CRD$AL_IDC_HDWR_SPRT_TABLE+24       ;
      00000000'  EF 00000000'  EF  9E 0007E    MOVAB   CRD$GT_IDC_EVRAD_0_PREP_ERR, -      ; 0695
           CRD$AL_IDC_HDWR_SPRT_TABLE+28       ;
      00000000'  EF 00000000'  EF  9E 00089    MOVAB   CRD$GT_IDC_EVRAD_1, -           ; 0697
           CRD$AL_IDC_HDWR_SPRT_TABLE+32       ;
      00000000'  EF 00000000'  EF  9E 00094    MOVAB   CRD$GT_IDC_EVRAD_1_SET_FLGS, -      ; 0698
           CRD$AL_IDC_HDWR_SPRT_TABLE+36       ;
      00000000'  EF 00000000'  EF  9E 0009F    MOVAB   CRD$GT_IDC_EVRAD_1_SET_EVNT, -      ; 0699
           CRD$AL_IDC_HDWR_SPRT_TABLE+40       ;
      00000000'  EF 00000000'  EF  9E 000AA    MOVAB   CRD$GT_IDC_EVRAD_1_STRT_QUAL, -     ; 0700
           CRD$AL_IDC_HDWR_SPRT_TABLE+48       ;
      00000000'  EF 00000000'  EF  9E 000B5    MOVAB   CRD$GT_IDC_EVRAD_1_DEV_NAME, -      ; 0701
           CRD$AL_IDC_HDWR_SPRT_TABLE+44       ;
      00000000'  EF 00000000'  EF  9E 000C0    MOVAB   CRD$GT_IDC_EVRAD_1_TST_STRT, -      ; 0702
           CRD$AL_IDC_HDWR_SPRT_TABLE+52       ;
      00000000'  EF 00000000'  EF  9E 000CB    MOVAB   CRD$GT_IDC_EVRAD_1_RUNTIME, -       ; 0703
           CRD$AL_IDC_HDWR_SPRT_TABLE+56       ;
      00000000'  EF 00000000'  EF  9E 000D6    MOVAB   CRD$GT_IDC_EVRAD_1_PREP_ERR, -      ; 0704
           CRD$AL_IDC_HDWR_SPRT_TABLE+60       ;
      00000000'  EF 00000000'  EF  9E 000E1    MOVAB   CRD$GT_IDC_EVDLB, -             ; 0706
           CRD$AL_IDC_HDWR_SPRT_TABLE+64       ;
      00000000'  EF 00000000'  EF  9E 000EC    MOVAB   CRD$GT_IDC_EVDLB_SET_FLGS, -        ; 0707
           CRD$AL_IDC_HDWR_SPRT_TABLE+68       ;
```

```
                00000000'  EF 00000000'  EF  9E 000F7    MOVAB   CRD$GT_IDC_EVDLB_SET_EVNT, -        ; 0708
                     CRD$AL_IDC_HDWR_SPRT_TABLE+72          ;
                00000000'  EF 00000000'  EF  9E 00102    MOVAB   CRD$GT_IDC_EVDLB_STRT_QUAL, -       ; 0709
                     CRD$AL_IDC_HDWR_SPRT_TABLE+80          ;
                00000000'  EF 00000000'  EF  9E 0010D    MOVAB   CRD$GT_IDC_EVDLB_DEV_NAME, -        ; 0710
                     CRD$AL_IDC_HDWR_SPRT_TABLE+76          ;
                00000000'  EF 00000000'  EF  9E 00118    MOVAB   CRD$GT_IDC_EVDLB_TST_STRT, -        ; 0711
                     CRD$AL_IDC_HDWR_SPRT_TABLE+84          ;
                00000000'  EF 00000000'  EF  9E 00123    MOVAB   CRD$GT_IDC_EVDLB_RUNTIME, -         ; 0712
                     CRD$AL_IDC_HDWR_SPRT_TABLE+88          ;
                00000000'  EF 00000000'  EF  9E 0012E    MOVAB   CRD$GT_IDC_EVDLB_PREP_ERR, -        ; 0713
                     CRD$AL_IDC_HDWR_SPRT_TABLE+92          ;
                00000000'  EF 00000000'  EF  9E 00139    MOVAB   CRD$GT_IDC_EVDLC, -         ; 0715
                     CRD$AL_IDC_HDWR_SPRT_TABLE+96          ;
                00000000'  EF 00000000'  EF  9E 00144    MOVAB   CRD$GT_IDC_EVDLC_SET_FLGS, -        ; 0716
                     CRD$AL_IDC_HDWR_SPRT_TABLE+100         ;
                00000000'  EF 00000000'  EF  9E 0014F    MOVAB   CRD$GT_IDC_EVDLC_SET_EVNT, -        ; 0717
                     CRD$AL_IDC_HDWR_SPRT_TABLE+104         ;
                00000000'  EF 00000000'  EF  9E 0015A    MOVAB   CRD$GT_IDC_EVDLC_STRT_QUAL, -       ; 0718
                     CRD$AL_IDC_HDWR_SPRT_TABLE+112         ;
                00000000'  EF 00000000'  EF  9E 00165    MOVAB   CRD$GT_IDC_EVDLC_DEV_NAME, -        ; 0719
                     CRD$AL_IDC_HDWR_SPRT_TABLE+108         ;
                00000000'  EF 00000000'  EF  9E 00170    MOVAB   CRD$GT_IDC_EVDLC_TST_STRT, -        ; 0720
                     CRD$AL_IDC_HDWR_SPRT_TABLE+116         ;
                00000000'  EF 00000000'  EF  9E 0017B    MOVAB   CRD$GT_IDC_EVDLC_RUNTIME, -         ; 0721
                     CRD$AL_IDC_HDWR_SPRT_TABLE+120         ;
                00000000'  EF 00000000'  EF  9E 00186    MOVAB   CRD$GT_IDC_EVDLC_PREP_ERR, -        ; 0722
                     CRD$AL_IDC_HDWR_SPRT_TABLE+124         ;
                00000000'  EF 00000000'  EF  9E 00191    MOVAB   CRD$GT_IDC_EVDLD, -         ; 0724
                     CRD$AL_IDC_HDWR_SPRT_TABLE+128         ;
                00000000'  EF 00000000'  EF  9E 0019C    MOVAB   CRD$GT_IDC_EVDLD_SET_FLGS, -        ; 0725
                     CRD$AL_IDC_HDWR_SPRT_TABLE+132         ;
                00000000'  EF 00000000'  EF  9E 001A7    MOVAB   CRD$GT_IDC_EVDLD_SET_EVNT, -        ; 0726
                     CRD$AL_IDC_HDWR_SPRT_TABLE+136         ;
                00000000'  EF 00000000'  EF  9E 001B2    MOVAB   CRD$GT_IDC_EVDLD_STRT_QUAL, -       ; 0727
                     CRD$AL_IDC_HDWR_SPRT_TABLE+144         ;
                00000000'  EF 00000000'  EF  9E 001BD    MOVAB   CRD$GT_IDC_EVDLD_DEV_NAME, -        ; 0728
                     CRD$AL_IDC_HDWR_SPRT_TABLE+140         ;
                00000000'  EF 00000000'  EF  9E 001C8    MOVAB   CRD$GT_IDC_EVDLD_TST_STRT, -        ; 0729
                     CRD$AL_IDC_HDWR_SPRT_TABLE+148         ;
                00000000'  EF 00000000'  EF  9E 001D3    MOVAB   CRD$GT_IDC_EVDLD_RUNTIME, -         ; 0730
                     CRD$AL_IDC_HDWR_SPRT_TABLE+152         ;
                00000000'  EF 00000000'  EF  9E 001DE    MOVAB   CRD$GT_IDC_EVDLD_PREP_ERR, -        ; 0731
                     CRD$AL_IDC_HDWR_SPRT_TABLE+156         ;
            52  D4 001E9     CLRL    DIAG                ; 0743
            51  D4 001EB 6$:   CLRL    ENTRY               ; 0744
            50    6241  DE 001ED 7$:   MOVAL   (DIAG)[ENTRY], R0          ; 0746
            03      52  D1 001F1    CMPL    DIAG, #3            ;
            05  18 001F4    BGEQ    8$                  ;
            08      51  D1 001F6    CMPL    ENTRY, #8           ;
            05  19 001F9    BLSS    9$                  ;
            50      01  CE 001FB 8$:   MNEGL   #1, R0            ;
            08  11 001FE    BRB     10$                 ;
            50 00000000'EF40  9E 00200 9$:   MOVAB   CRD$AL_LESI_HDWR_SPRT_TABLE[R0], R0       ;
            60  D4 00208 10$:   CLRL    (R0)                ;
```

```
                DF        51      07  F3 0020A    AOBLEQ  #7, ENTRY, 7$          ; 0744
  FFD3           52        20 00000040   8F  F1 0020E    ACBL    #64, #32, DIAG, 6$      ;
        00000000'  EF 00000000'  EF  9E 00218    MOVAB   CRD$GT_LESI_EVRMB_0, -          ; 0753
             CRD$AL_LESI_HDWR_SPRT_TABLE          ;
        00000000'  EF 00000000'  EF  9E 00223    MOVAB   CRD$GT_LESI_EVRMB_0_SET_FLGS, -        ; 0754
             CRD$AL_LESI_HDWR_SPRT_TABLE+4        ;
        00000000'  EF 00000000'  EF  9E 0022E    MOVAB   CRD$GT_LESI_EVRMB_0_SET_EVNT, -        ; 0755
             CRD$AL_LESI_HDWR_SPRT_TABLE+8        ;
        00000000'  EF 00000000'  EF  9E 00239    MOVAB   CRD$GT_LESI_EVRMB_0_STRT_QUAL, -        ; 0756
             CRD$AL_LESI_HDWR_SPRT_TABLE+16       ;
        00000000'  EF 00000000'  EF  9E 00244    MOVAB   CRD$GT_LESI_EVRMB_0_DEV_NAME, -        ; 0757
             CRD$AL_LESI_HDWR_SPRT_TABLE+12       ;
        00000000'  EF 00000000'  EF  9E 0024F    MOVAB   CRD$GT_LESI_EVRMB_0_TST_STRT, -        ; 0758
             CRD$AL_LESI_HDWR_SPRT_TABLE+20       ;
        00000000'  EF 00000000'  EF  9E 0025A    MOVAB   CRD$GT_LESI_EVRMB_0_RUNTIME, -        ; 0759
             CRD$AL_LESI_HDWR_SPRT_TABLE+24       ;
        00000000'  EF 00000000'  EF  9E 00265    MOVAB   CRD$GT_LESI_EVRMB_0_PREP_ERR, -        ; 0760
             CRD$AL_LESI_HDWR_SPRT_TABLE+28       ;
        00000000'  EF 00000000'  EF  9E 00270    MOVAB   CRD$GT_LESI_EVRMB_1, -          ; 0762
             CRD$AL_LESI_HDWR_SPRT_TABLE+32       ;
        00000000'  EF 00000000'  EF  9E 0027B    MOVAB   CRD$GT_LESI_EVRMB_1_SET_FLGS, -        ; 0763
             CRD$AL_LESI_HDWR_SPRT_TABLE+36       ;
        00000000'  EF 00000000'  EF  9E 00286    MOVAB   CRD$GT_LESI_EVRMB_1_SET_EVNT, -        ; 0764
             CRD$AL_LESI_HDWR_SPRT_TABLE+40       ;
        00000000'  EF 00000000'  EF  9E 00291    MOVAB   CRD$GT_LESI_EVRMB_1_STRT_QUAL, -        ; 0765
             CRD$AL_LESI_HDWR_SPRT_TABLE+48       ;
        00000000'  EF 00000000'  EF  9E 0029C    MOVAB   CRD$GT_LESI_EVRMB_1_DEV_NAME, -        ; 0766
             CRD$AL_LESI_HDWR_SPRT_TABLE+44       ;
        00000000'  EF 00000000'  EF  9E 002A7    MOVAB   CRD$GT_LESI_EVRMB_1_TST_STRT, -        ; 0767
             CRD$AL_LESI_HDWR_SPRT_TABLE+52       ;
        00000000'  EF 00000000'  EF  9E 002B2    MOVAB   CRD$GT_LESI_EVRMB_1_RUNTIME, -        ; 0768
             CRD$AL_LESI_HDWR_SPRT_TABLE+56       ;
        00000000'  EF 00000000'  EF  9E 002BD    MOVAB   CRD$GT_LESI_EVRMB_1_PREP_ERR, -        ; 0769
             CRD$AL_LESI_HDWR_SPRT_TABLE+60       ;
            52  D4 002C8    CLRL    DIAG            ; 0793
            51  D4 002CA 11$:   CLRL    ENTRY           ; 0794
            50    6241  DE 002CC 12$:   MOVAL   (DIAG)[ENTRY], R0        ; 0796
            04      52  D1 002D0    CMPL    DIAG, #4          ;
            05  18 002D3    BGEQ    13$             ;
            08      51  D1 002D5    CMPL    ENTRY, #8         ;
            05  19 002D8    BLSS    14$             ;
            50      01  CE 002DA 13$:   MNEGL   #1, R0          ;
            08  11 002DD    BRB     15$             ;
            50 00000000'EF40  9E 002DF 14$:   MOVAB   CRD$AL_UDA_0_HDWR_SPRT_TABLE[R0], R0      ;
            60  D4 002E7 15$:   CLRL    (R0)            ;
            DF        51      07  F3 002E9    AOBLEQ  #7, ENTRY, 12$          ; 0794
  FFD3       52        20 00000060   8F  F1 002ED    ACBL    #96, #32, DIAG, 11$       ;
        00000000'  EF 00000000'  EF  9E 002F7    MOVAB   CRD$GT_UDA_0_EVRLA_0, -         ; 0803
             CRD$AL_UDA_0_HDWR_SPRT_TABLE         ;
        00000000'  EF 00000000'  EF  9E 00302    MOVAB   CRD$GT_UDA_0_EVRLA_0_SET_FLGS, -       ; 0804
             CRD$AL_UDA_0_HDWR_SPRT_TABLE+4       ;
        00000000'  EF 00000000'  EF  9E 0030D    MOVAB   CRD$GT_UDA_0_EVRLA_0_SET_EVNT, -       ; 0805
             CRD$AL_UDA_0_HDWR_SPRT_TABLE+8       ;
        00000000'  EF 00000000'  EF  9E 00318    MOVAB   CRD$GT_UDA_0_EVRLA_0_STRT_QUAL, -       ; 0806
             CRD$AL UDA_0_HDWR_SPRT_TABLE+16      ;
```

```
00000000'  EF 00000000'  EF  9E 00323    MOVAB   CRD$GT_UDA_0_EVRLA_0_DEV_NAME, -       ; 0807
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+12
                                          ;
00000000'  EF 00000000'  EF  9E 0032E    MOVAB   CRD$GT_UDA_0_EVRLA_0_TST_STRT, -       ; 0808
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+20
                                          ;
00000000'  EF 00000000'  EF  9E 00339    MOVAB   CRD$GT_UDA_0_EVRLA_0_RUNTIME, -        ; 0809
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+24
                                          ;
00000000'  EF 00000000'  EF  9E 00344    MOVAB   CRD$GT_UDA_0_EVRLA_0_PREP_ERR, -       ; 0810
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+28
                                          ;
00000000'  EF 00000000'  EF  9E 0034F    MOVAB   CRD$GT_UDA_0_EVDLB, -          ; 0812
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+32
                                          ;
00000000'  EF 00000000'  EF  9E 0035A    MOVAB   CRD$GT_UDA_0_EVDLB_SET_FLGS, -       ; 0813
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+36
                                          ;
00000000'  EF 00000000'  EF  9E 00365    MOVAB   CRD$GT_UDA_0_EVDLB_SET_EVNT, -       ; 0814
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+40
                                          ;
00000000'  EF 00000000'  EF  9E 00370    MOVAB   CRD$GT_UDA_0_EVDLB_STRT_QUAL, -       ; 0815
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+48
                                          ;
00000000'  EF 00000000'  EF  9E 0037B    MOVAB   CRD$GT_UDA_0_EVDLB_DEV_NAME, -       ; 0816
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+44
                                          ;
00000000'  EF 00000000'  EF  9E 00386    MOVAB   CRD$GT_UDA_0_EVDLB_TST_STRT, -       ; 0817
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+52
                                          ;
00000000'  EF 00000000'  EF  9E 00391    MOVAB   CRD$GT_UDA_0_EVDLB_RUNTIME, -        ; 0818
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+56
                                          ;
00000000'  EF 00000000'  EF  9E 0039C    MOVAB   CRD$GT_UDA_0_EVDLB_PREP_ERR, -       ; 0819
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+60
                                          ;
00000000'  EF 00000000'  EF  9E 003A7    MOVAB   CRD$GT_UDA_0_EVDLC, -          ; 0821
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+64
                                          ;
00000000'  EF 00000000'  EF  9E 003B2    MOVAB   CRD$GT_UDA_0_EVDLC_SET_FLGS, -       ; 0822
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+68
                                          ;
00000000'  EF 00000000'  EF  9E 003BD    MOVAB   CRD$GT_UDA_0_EVDLC_SET_EVNT, -       ; 0823
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+72
                                          ;
00000000'  EF 00000000'  EF  9E 003C8    MOVAB   CRD$GT_UDA_0_EVDLC_STRT_QUAL, -       ; 0824
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+80
                                          ;
00000000'  EF 00000000'  EF  9E 003D3    MOVAB   CRD$GT_UDA_0_EVDLC_DEV_NAME, -       ; 0825
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+76
                                          ;
00000000'  EF 00000000'  EF  9E 003DE    MOVAB   CRD$GT_UDA_0_EVDLC_TST_STRT, -       ; 0826
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+84
                                          ;
00000000'  EF 00000000'  EF  9E 003E9    MOVAB   CRD$GT_UDA_0_EVDLC_RUNTIME, -        ; 0827
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+88
                                          ;
00000000'  EF 00000000'  EF  9E 003F4    MOVAB   CRD$GT_UDA_0_EVDLC_PREP_ERR, -       ; 0828
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+92
                                          ;
00000000'  EF 00000000'  EF  9E 003FF    MOVAB   CRD$GT_UDA_0_EVDLD, -          ; 0830
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+96
                                          ;
00000000'  EF 00000000'  EF  9E 0040A    MOVAB   CRD$GT_UDA_0_EVDLD_SET_FLGS, -       ; 0831
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+100
                                          ;
00000000'  EF 00000000'  EF  9E 00415    MOVAB   CRD$GT_UDA_0_EVDLD_SET_EVNT, -       ; 0832
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+104
                                          ;
00000000'  EF 00000000'  EF  9E 00420    MOVAB   CRD$GT_UDA_0_EVDLD_STRT_QUAL, -       ; 0833
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+112
                                          ;
00000000'  EF 00000000'  EF  9E 0042B    MOVAB   CRD$GT_UDA_0_EVDLD_DEV_NAME, -       ; 0834
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+108
                                          ;
00000000'  EF 00000000'  EF  9E 00436    MOVAB   CRD$GT_UDA_0_EVDLD_TST_STRT, -       ; 0835
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+116
                                          ;
00000000'  EF 00000000'  EF  9E 00441    MOVAB   CRD$GT_UDA_0_EVDLD_RUNTIME, -        ; 0836
     CRD$AL_UDA_0_HDWR_SPRT_TABLE+120
                                          ;
00000000'  EF 00000000'  EF  9E 0044C    MOVAB   CRD$GT_UDA_0_EVDLD_PREP_ERR, -       ; 0837
```

```
            CRD$AL_UDA_0_HDWR_SPRT_TABLE+124        ;
           52  D4 00457    CLRL    DIAG            ; 0852
           51  D4 00459 16$:   CLRL    ENTRY           ; 0853
           50    6241  DE 0045B 17$:   MOVAL   (DIAG)[ENTRY], R0         ; 0855
           05      52  D1 0045F    CMPL    DIAG, #5          ;
           05  18 00462    BGEQ    18$             ;
           08      51  D1 00464    CMPL    ENTRY, #8         ;
           05  19 00467    BLSS    19$             ;
           50      01  CE 00469 18$:   MNEGL   #1, R0          ;
           08  11 0046C    BRB     20$             ;
           50 00000000'EF40  9E 0046E 19$:   MOVAB   CRD$AL_UDA_1_HDWR_SPRT_TABLE[R0], R0       ;
           60  D4 00476 20$:   CLRL    (R0)            ;
           DF        51      07  F3 00478    AOBLEQ  #7, ENTRY, 17$          ; 0853
  FFD3          52        20 00000080   8F  F1 0047C    ACBL    #128, #32, DIAG, 16$        ;
       00000000'  EF 00000000'  EF  9E 00486    MOVAB   CRD$GT_UDA_1_EVRLA_0, -         ; 0862
            CRD$AL_UDA_1_HDWR_SPRT_TABLE        ;
       00000000'  EF 00000000'  EF  9E 00491    MOVAB   CRD$GT_UDA_1_EVRLA_0_SET_FLGS, -        ; 0863
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+4      ;
       00000000'  EF 00000000'  EF  9E 0049C    MOVAB   CRD$GT_UDA_1_EVRLA_0_SET_EVNT, -        ; 0864
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+8      ;
       00000000'  EF 00000000'  EF  9E 004A7    MOVAB   CRD$GT_UDA_1_EVRLA_0_STRT_QUAL, -       ; 0865
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+16     ;
       00000000'  EF 00000000'  EF  9E 004B2    MOVAB   CRD$GT_UDA_1_EVRLA_0_DEV_NAME, -        ; 0866
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+12     ;
       00000000'  EF 00000000'  EF  9E 004BD    MOVAB   CRD$GT_UDA_1_EVRLA_0_TST_STRT, -        ; 0867
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+20     ;
       00000000'  EF 00000000'  EF  9E 004C8    MOVAB   CRD$GT_UDA_1_EVRLA_0_RUNTIME, -         ; 0868
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+24     ;
       00000000'  EF 00000000'  EF  9E 004D3    MOVAB   CRD$GT_UDA_1_EVRLA_0_PREP_ERR, -        ; 0869
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+28     ;
       00000000'  EF 00000000'  EF  9E 004DE    MOVAB   CRD$GT_UDA_1_EVRLA_1, -         ; 0871
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+32     ;
       00000000'  EF 00000000'  EF  9E 004E9    MOVAB   CRD$GT_UDA_1_EVRLA_1_SET_FLGS, -        ; 0872
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+36     ;
       00000000'  EF 00000000'  EF  9E 004F4    MOVAB   CRD$GT_UDA_1_EVRLA_1_SET_EVNT, -        ; 0873
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+40     ;
       00000000'  EF 00000000'  EF  9E 004FF    MOVAB   CRD$GT_UDA_1_EVRLA_1_STRT_QUAL, -       ; 0874
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+48     ;
       00000000'  EF 00000000'  EF  9E 0050A    MOVAB   CRD$GT_UDA_1_EVRLA_1_DEV_NAME, -        ; 0875
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+44     ;
       00000000'  EF 00000000'  EF  9E 00515    MOVAB   CRD$GT_UDA_1_EVRLA_1_TST_STRT, -        ; 0876
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+52     ;
       00000000'  EF 00000000'  EF  9E 00520    MOVAB   CRD$GT_UDA_1_EVRLA_1_RUNTIME, -         ; 0877
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+56     ;
       00000000'  EF 00000000'  EF  9E 0052B    MOVAB   CRD$GT_UDA_1_EVRLA_1_PREP_ERR, -        ; 0878
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+60     ;
       00000000'  EF 00000000'  EF  9E 00536    MOVAB   CRD$GT_UDA_1_EVDLB, -           ; 0880
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+64     ;
       00000000'  EF 00000000'  EF  9E 00541    MOVAB   CRD$GT_UDA_1_EVDLB_SET_FLGS, -          ; 0881
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+68     ;
       00000000'  EF 00000000'  EF  9E 0054C    MOVAB   CRD$GT_UDA_1_EVDLB_SET_EVNT, -          ; 0882
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+72     ;
       00000000'  EF 00000000'  EF  9E 00557    MOVAB   CRD$GT_UDA_1_EVDLB_STRT_QUAL, -         ; 0883
            CRD$AL_UDA_1_HDWR_SPRT_TABLE+80     ;
       00000000'  EF 00000000'  EF  9E 00562    MOVAB   CRD$GT_UDA_1_EVDLB_DEV_NAME, -          ; 0884
```

```
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+76      ;
        00000000'  EF 00000000'  EF  9E 0056D    MOVAB   CRD$GT_UDA_1_EVDLB_TST_STRT, -       ; 0885
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+84      ;
        00000000'  EF 00000000'  EF  9E 00578    MOVAB   CRD$GT_UDA_1_EVDLB_RUNTIME, -        ; 0886
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+88      ;
        00000000'  EF 00000000'  EF  9E 00583    MOVAB   CRD$GT_UDA_1_EVDLB_PREP_ERR, -       ; 0887
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+92      ;
        00000000'  EF 00000000'  EF  9E 0058E    MOVAB   CRD$GT_UDA_1_EVDLC, -          ; 0889
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+96      ;
        00000000'  EF 00000000'  EF  9E 00599    MOVAB   CRD$GT_UDA_1_EVDLC_SET_FLGS, -       ; 0890
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+100     ;
        00000000'  EF 00000000'  EF  9E 005A4    MOVAB   CRD$GT_UDA_1_EVDLC_SET_EVNT, -       ; 0891
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+104     ;
        00000000'  EF 00000000'  EF  9E 005AF    MOVAB   CRD$GT_UDA_1_EVDLC_STRT_QUAL, -       ; 0892
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+112     ;
        00000000'  EF 00000000'  EF  9E 005BA    MOVAB   CRD$GT_UDA_1_EVDLC_DEV_NAME, -       ; 0893
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+108     ;
        00000000'  EF 00000000'  EF  9E 005C5    MOVAB   CRD$GT_UDA_1_EVDLC_TST_STRT, -       ; 0894
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+116     ;
        00000000'  EF 00000000'  EF  9E 005D0    MOVAB   CRD$GT_UDA_1_EVDLC_RUNTIME, -        ; 0895
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+120     ;
        00000000'  EF 00000000'  EF  9E 005DB    MOVAB   CRD$GT_UDA_1_EVDLC_PREP_ERR, -       ; 0896
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+124     ;
        00000000'  EF 00000000'  EF  9E 005E6    MOVAB   CRD$GT_UDA_1_EVDLD, -          ; 0898
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+128     ;
        00000000'  EF 00000000'  EF  9E 005F1    MOVAB   CRD$GT_UDA_1_EVDLD_SET_FLGS, -       ; 0899
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+132     ;
        00000000'  EF 00000000'  EF  9E 005FC    MOVAB   CRD$GT_UDA_1_EVDLD_SET_EVNT, -       ; 0900
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+136     ;
        00000000'  EF 00000000'  EF  9E 00607    MOVAB   CRD$GT_UDA_1_EVDLD_STRT_QUAL, -       ; 0901
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+144     ;
        00000000'  EF 00000000'  EF  9E 00612    MOVAB   CRD$GT_UDA_1_EVDLD_DEV_NAME, -       ; 0902
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+140     ;
        00000000'  EF 00000000'  EF  9E 0061D    MOVAB   CRD$GT_UDA_1_EVDLD_TST_STRT, -       ; 0903
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+148     ;
        00000000'  EF 00000000'  EF  9E 00628    MOVAB   CRD$GT_UDA_1_EVDLD_RUNTIME, -        ; 0904
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+152     ;
        00000000'  EF 00000000'  EF  9E 00633    MOVAB   CRD$GT_UDA_1_EVDLD_PREP_ERR, -       ; 0905
                CRD$AL_UDA_1_HDWR_SPRT_TABLE+156     ;
            52  D4 0063E     CLRL    DIAG            ; 0918
            51  D4 00640 21$:   CLRL    ENTRY           ; 0919
            50    6241  DE 00642 22$:   MOVAL   (DIAG)[ENTRY], R0         ; 0921
            04      52  D1 00646     CMPL    DIAG, #4          ;
            05  18 00649     BGEQ    23$               ;
            08      51  D1 0064B     CMPL    ENTRY, #8         ;
            05  19 0064E     BLSS    24$               ;
            50      01  CE 00650 23$:   MNEGL   #1, R0          ;
            08  11 00653     BRB     25$               ;
            50 00000000'EF40  9E 00655 24$:   MOVAB   CRD$AL_UDA_2_HDWR_SPRT_TABLE[R0], R0       ;
            60  D4 0065D 25$:   CLRL    (R0)              ;
            DF        51      07  F3 0065F    AOBLEQ  #7, ENTRY, 22$          ; 0919
FFD3          52        20 00000060   8F  F1 00663    ACBL    #96, #32, DIAG, 21$         ;
        00000000'  EF 00000000'  EF  9E 0066D    MOVAB   CRD$GT_UDA_2_EVRLA_0, -         ; 0928
                CRD$AL_UDA_2_HDWR_SPRT_TABLE         ;
        00000000'  EF 00000000'  EF  9E 00678    MOVAB   CRD$GT_UDA_2_EVRLA_0_SET_FLGS, -       ; 0929
```

```
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+4
                                        ;
00000000'  EF 00000000'  EF  9E 00683    MOVAB   CRD$GT_UDA_2_EVRLA_0_SET_EVNT, -       ; 0930
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+8
                                        ;
00000000'  EF 00000000'  EF  9E 0068E    MOVAB   CRD$GT_UDA_2_EVRLA_0_STRT_QUAL, -       ; 0931
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+16
                                        ;
00000000'  EF 00000000'  EF  9E 00699    MOVAB   CRD$GT_UDA_2_EVRLA_0_DEV_NAME, -        ; 0932
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+12
                                        ;
00000000'  EF 00000000'  EF  9E 006A4    MOVAB   CRD$GT_UDA_2_EVRLA_0_TST_STRT, -        ; 0933
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+20
                                        ;
00000000'  EF 00000000'  EF  9E 006AF    MOVAB   CRD$GT_UDA_2_EVRLA_0_RUNTIME, -        ; 0934
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+24
                                        ;
00000000'  EF 00000000'  EF  9E 006BA    MOVAB   CRD$GT_UDA_2_EVRLA_0_PREP_ERR, -        ; 0935
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+28
                                        ;
00000000'  EF 00000000'  EF  9E 006C5    MOVAB   CRD$GT_UDA_2_EVDLB, -          ; 0937
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+32
                                        ;
00000000'  EF 00000000'  EF  9E 006D0    MOVAB   CRD$GT_UDA_2_EVDLB_SET_FLGS, -        ; 0938
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+36
                                        ;
00000000'  EF 00000000'  EF  9E 006DB    MOVAB   CRD$GT_UDA_2_EVDLB_SET_EVNT, -        ; 0939
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+40
                                        ;
00000000'  EF 00000000'  EF  9E 006E6    MOVAB   CRD$GT_UDA_2_EVDLB_STRT_QUAL, -       ; 0940
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+48
                                        ;
00000000'  EF 00000000'  EF  9E 006F1    MOVAB   CRD$GT_UDA_2_EVDLB_DEV_NAME, -        ; 0941
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+44
                                        ;
00000000'  EF 00000000'  EF  9E 006FC    MOVAB   CRD$GT_UDA_2_EVDLB_TST_STRT, -        ; 0942
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+52
                                        ;
00000000'  EF 00000000'  EF  9E 00707    MOVAB   CRD$GT_UDA_2_EVDLB_RUNTIME, -        ; 0943
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+56
                                        ;
00000000'  EF 00000000'  EF  9E 00712    MOVAB   CRD$GT_UDA_2_EVDLB_PREP_ERR, -        ; 0944
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+60
                                        ;
00000000'  EF 00000000'  EF  9E 0071D    MOVAB   CRD$GT_UDA_2_EVDLC, -          ; 0946
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+64
                                        ;
00000000'  EF 00000000'  EF  9E 00728    MOVAB   CRD$GT_UDA_2_EVDLC_SET_FLGS, -        ; 0947
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+68
                                        ;
00000000'  EF 00000000'  EF  9E 00733    MOVAB   CRD$GT_UDA_2_EVDLC_SET_EVNT, -        ; 0948
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+72
                                        ;
00000000'  EF 00000000'  EF  9E 0073E    MOVAB   CRD$GT_UDA_2_EVDLC_STRT_QUAL, -       ; 0949
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+80
                                        ;
00000000'  EF 00000000'  EF  9E 00749    MOVAB   CRD$GT_UDA_2_EVDLC_DEV_NAME, -        ; 0950
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+76
                                        ;
00000000'  EF 00000000'  EF  9E 00754    MOVAB   CRD$GT_UDA_2_EVDLC_TST_STRT, -        ; 0951
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+84
                                        ;
00000000'  EF 00000000'  EF  9E 0075F    MOVAB   CRD$GT_UDA_2_EVDLC_RUNTIME, -        ; 0952
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+88
                                        ;
00000000'  EF 00000000'  EF  9E 0076A    MOVAB   CRD$GT_UDA_2_EVDLC_PREP_ERR, -        ; 0953
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+92
                                        ;
00000000'  EF 00000000'  EF  9E 00775    MOVAB   CRD$GT_UDA_2_EVDLD, -          ; 0955
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+96
                                        ;
00000000'  EF 00000000'  EF  9E 00780    MOVAB   CRD$GT_UDA_2_EVDLD_SET_FLGS, -        ; 0956
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+100
                                        ;
00000000'  EF 00000000'  EF  9E 0078B    MOVAB   CRD$GT_UDA_2_EVDLD_SET_EVNT, -        ; 0957
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+104
                                        ;
00000000'  EF 00000000'  EF  9E 00796    MOVAB   CRD$GT_UDA_2_EVDLD_STRT_QUAL, -       ; 0958
                CRD$AL_UDA_2_HDWR_SPRT_TABLE+112
                                        ;
00000000'  EF 00000000'  EF  9E 007A1    MOVAB   CRD$GT_UDA_2_EVDLD_DEV_NAME, -        ; 0959
                CRD$AL_UDA_2 HDWR_SPRT_TABLE+108
                                        ;
```

```
        00000000'  EF 00000000'  EF  9E 007AC     MOVAB   CRD$GT_UDA_2_EVDLD_TST_STRT, -       ; 0960
             CRD$AL_UDA_2_HDWR_SPRT_TABLE+116       ;
        00000000'  EF 00000000'  EF  9E 007B7     MOVAB   CRD$GT_UDA_2_EVDLD_RUNTIME, -        ; 0961
             CRD$AL_UDA_2_HDWR_SPRT_TABLE+120       ;
        00000000'  EF 00000000'  EF  9E 007C2     MOVAB   CRD$GT_UDA_2_EVDLD_PREP_ERR, -       ; 0962
             CRD$AL_UDA_2_HDWR_SPRT_TABLE+124       ;
            52  D4 007CD     CLRL    DIAG            ; 0977
            51  D4 007CF 26$:   CLRL    ENTRY           ; 0978
            50    6241  DE 007D1 27$:   MOVAL   (DIAG)[ENTRY], R0         ; 0980
            05      52  D1 007D5    CMPL    DIAG, #5          ;
            05  18 007D8    BGEQ    28$             ;
            08      51  D1 007DA    CMPL    ENTRY, #8         ;
            05  19 007DD    BLSS    29$             ;
            50      01  CE 007DF 28$:   MNEGL   #1, R0          ;
            08  11 007E2    BRB     30$             ;
            50 00000000'EF40  9E 007E4 29$:   MOVAB   CRD$AL_UDA_3_HDWR_SPRT_TABLE[R0], R0       ;
            60  D4 007EC 30$:   CLRL    (R0)            ;
            DF      51      07  F3 007EE    AOBLEQ  #7, ENTRY, 27$          ; 0978
      FFD3      52      20 00000080   8F  F1 007F2    ACBL    #128, #32, DIAG, 26$        ;
        00000000'  EF 00000000'  EF  9E 007FC     MOVAB   CRD$GT_UDA_3_EVRLA_0, -          ; 0987
             CRD$AL_UDA_3_HDWR_SPRT_TABLE           ;
        00000000'  EF 00000000'  EF  9E 00807     MOVAB   CRD$GT_UDA_3_EVRLA_0_SET_FLGS, -       ; 0988
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+4         ;
        00000000'  EF 00000000'  EF  9E 00812     MOVAB   CRD$GT_UDA_3_EVRLA_0_SET_EVNT, -       ; 0989
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+8         ;
        00000000'  EF 00000000'  EF  9E 0081D     MOVAB   CRD$GT_UDA_3_EVRLA_0_STRT_QUAL, -       ; 0990
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+16        ;
        00000000'  EF 00000000'  EF  9E 00828     MOVAB   CRD$GT_UDA_3_EVRLA_0_DEV_NAME, -       ; 0991
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+12        ;
        00000000'  EF 00000000'  EF  9E 00833     MOVAB   CRD$GT_UDA_3_EVRLA_0_TST_STRT, -       ; 0992
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+20        ;
        00000000'  EF 00000000'  EF  9E 0083E     MOVAB   CRD$GT_UDA_3_EVRLA_0_RUNTIME, -       ; 0993
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+24        ;
        00000000'  EF 00000000'  EF  9E 00849     MOVAB   CRD$GT_UDA_3_EVRLA_0_PREP_ERR, -       ; 0994
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+28        ;
        00000000'  EF 00000000'  EF  9E 00854     MOVAB   CRD$GT_UDA_3_EVRLA_1, -          ; 0996
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+32        ;
        00000000'  EF 00000000'  EF  9E 0085F     MOVAB   CRD$GT_UDA_3_EVRLA_1_SET_FLGS, -       ; 0997
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+36        ;
        00000000'  EF 00000000'  EF  9E 0086A     MOVAB   CRD$GT_UDA_3_EVRLA_1_SET_EVNT, -       ; 0998
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+40        ;
        00000000'  EF 00000000'  EF  9E 00875     MOVAB   CRD$GT_UDA_3_EVRLA_1_STRT_QUAL, -       ; 0999
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+48        ;
        00000000'  EF 00000000'  EF  9E 00880     MOVAB   CRD$GT_UDA_3_EVRLA_1_DEV_NAME, -       ; 1000
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+44        ;
        00000000'  EF 00000000'  EF  9E 0088B     MOVAB   CRD$GT_UDA_3_EVRLA_1_TST_STRT, -       ; 1001
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+52        ;
        00000000'  EF 00000000'  EF  9E 00896     MOVAB   CRD$GT_UDA_3_EVRLA_1_RUNTIME, -       ; 1002
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+56        ;
        00000000'  EF 00000000'  EF  9E 008A1     MOVAB   CRD$GT_UDA_3_EVRLA_1_PREP_ERR, -       ; 1003
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+60        ;
        00000000'  EF 00000000'  EF  9E 008AC     MOVAB   CRD$GT_UDA_3_EVDLB, -          ; 1005
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+64        ;
        00000000'  EF 00000000'  EF  9E 008B7     MOVAB   CRD$GT_UDA_3_EVDLB_SET_FLGS, -       ; 1006
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+68        ;
```

```
        00000000'  EF 00000000'  EF  9E 008C2    MOVAB   CRD$GT_UDA_3_EVDLB_SET_EVNT, -       ; 1007
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+72          ;
        00000000'  EF 00000000'  EF  9E 008CD    MOVAB   CRD$GT_UDA_3_EVDLB_STRT_QUAL, -       ; 1008
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+80          ;
        00000000'  EF 00000000'  EF  9E 008D8    MOVAB   CRD$GT_UDA_3_EVDLB_DEV_NAME, -       ; 1009
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+76          ;
        00000000'  EF 00000000'  EF  9E 008E3    MOVAB   CRD$GT_UDA_3_EVDLB_TST_STRT, -       ; 1010
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+84          ;
        00000000'  EF 00000000'  EF  9E 008EE    MOVAB   CRD$GT_UDA_3_EVDLB_RUNTIME, -       ; 1011
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+88          ;
        00000000'  EF 00000000'  EF  9E 008F9    MOVAB   CRD$GT_UDA_3_EVDLB_PREP_ERR, -       ; 1012
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+92          ;
        00000000'  EF 00000000'  EF  9E 00904    MOVAB   CRD$GT_UDA_3_EVDLC, -         ; 1014
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+96          ;
        00000000'  EF 00000000'  EF  9E 0090F    MOVAB   CRD$GT_UDA_3_EVDLC_SET_FLGS, -       ; 1015
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+100         ;
        00000000'  EF 00000000'  EF  9E 0091A    MOVAB   CRD$GT_UDA_3_EVDLC_SET_EVNT, -       ; 1016
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+104         ;
        00000000'  EF 00000000'  EF  9E 00925    MOVAB   CRD$GT_UDA_3_EVDLC_STRT_QUAL, -       ; 1017
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+112         ;
        00000000'  EF 00000000'  EF  9E 00930    MOVAB   CRD$GT_UDA_3_EVDLC_DEV_NAME, -       ; 1018
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+108         ;
        00000000'  EF 00000000'  EF  9E 0093B    MOVAB   CRD$GT_UDA_3_EVDLC_TST_STRT, -       ; 1019
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+116         ;
        00000000'  EF 00000000'  EF  9E 00946    MOVAB   CRD$GT_UDA_3_EVDLC_RUNTIME, -       ; 1020
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+120         ;
        00000000'  EF 00000000'  EF  9E 00951    MOVAB   CRD$GT_UDA_3_EVDLC_PREP_ERR, -       ; 1021
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+124         ;
        00000000'  EF 00000000'  EF  9E 0095C    MOVAB   CRD$GT_UDA_3_EVDLD, -         ; 1023
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+128         ;
        00000000'  EF 00000000'  EF  9E 00967    MOVAB   CRD$GT_UDA_3_EVDLD_SET_FLGS, -       ; 1024
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+132         ;
        00000000'  EF 00000000'  EF  9E 00972    MOVAB   CRD$GT_UDA_3_EVDLD_SET_EVNT, -       ; 1025
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+136         ;
        00000000'  EF 00000000'  EF  9E 0097D    MOVAB   CRD$GT_UDA_3_EVDLD_STRT_QUAL, -       ; 1026
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+144         ;
        00000000'  EF 00000000'  EF  9E 00988    MOVAB   CRD$GT_UDA_3_EVDLD_DEV_NAME, -       ; 1027
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+140         ;
        00000000'  EF 00000000'  EF  9E 00993    MOVAB   CRD$GT_UDA_3_EVDLD_TST_STRT, -       ; 1028
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+148         ;
        00000000'  EF 00000000'  EF  9E 0099E    MOVAB   CRD$GT_UDA_3_EVDLD_RUNTIME, -       ; 1029
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+152         ;
        00000000'  EF 00000000'  EF  9E 009A9    MOVAB   CRD$GT_UDA_3_EVDLD_PREP_ERR, -       ; 1030
             CRD$AL_UDA_3_HDWR_SPRT_TABLE+156         ;
         04 009B4    RET                 ; 1033

; Routine Size:  2485 bytes,    Routine Base:  CODE + 0000
```

```
;    1034  1 %SBTTL 'CRD$Print_Hardware_Support Routine'
;    1035  1
;    1036  1 GLOBAL ROUTINE CRD$Print_Hardware_Support : NOVALUE =
;    1037  1
;    1038  1 !++
;    1039  1 ! Functional Description:
;    1040  1 !
;    1041  1 ! Print the contents of the IDC hardware support table.       [04]
;    1042  1 !
;    1043  1 ! Formal Input Parameters:
;    1044  1 !
;    1045  1 ! None
;    1046  1 !
;    1047  1 ! Formal Output Parameters:
;    1048  1 !
;    1049  1 ! None
;    1050  1 !
;    1051  1 ! Side Effects:
;    1052  1 !
;    1053  1 ! None
;    1054  1 !
;    1055  1 ! Completion Codes:
;    1056  1 !
;    1057  1 ! None
;    1058  1 !--
```

```
;   1059  2 BEGIN     ! Routine CRD$Print_Hardware_Support
;   1060  2  BIND
;   1061  2   Out_HS_Header  = $ASCIC ('!/****************** The IDC Hardware Support Table ******************!/'),
;   1062  2   Out_HS_Table = $ASCIC ('Hardware_Support [!1UL, !1UL]: .!AC.!/'),
;   1063  2   T_Nothing_There = $ASCIC ('**********************');
;   1064  2
;   1065  2  $CRD_Print (Out_HS_Header);
;   1066  2
;   1067  2  INCR HS_Diag FROM 0 TO (CRD$K_IDC_Numb_Diagnostics - 1) DO
;   1068  3   BEGIN
;   1069  3   INCR HS_Entry FROM 0 TO (CRD$K_Numb_Support_Entries - 1) DO
;   1070  4    BEGIN
; P 1071  4    $CRD_Print (
; P 1072  4     Out_HS_Table,
; P 1073  4     .HS_Diag,
; P 1074  4     .HS_Entry,
; P 1075  4     (
; P 1076  4      BIND
; P 1077  4       Ascic_String =
; P 1078  4        (.CRD$AL_IDC_Hdwr_Sprt_Table [.HS_Diag, .HS_Entry])
; P 1079  4        : VECTOR [, BYTE];
; P 1080  4
; P 1081  4      !+
; P 1082  4      ! If the count byte is zero, print nothing is there.
; P 1083  4      ! Else print the Acsic string.
; P 1084  4      !-
; P 1085  4
; P 1086  4      IF (.Ascic_String [0] EQL 0) THEN
; P 1087  4       T_Nothing_There
; P 1088  4      ELSE
; P 1089  4       Ascic_String
; P 1090  4     )
;   1091  4     );
;   1092  3    END;
;   1093  3
;   1094  3   $CRD_Print ($ASCIC ('!/'));
;   1095  3      ! Blank line between rows
;   1096  2   END;
;   1097  2
;   1098  2  RETURN;
;   1099  1 END;     ! Routine CRD$Print_Hardware_Support


            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2F  21  48  00827 P.AHY:  .ASCII  \H!/****************** The IDC Hardware S\  ;
20  43  44  49  20  65  68  54  20  2A  2A  2A  2A  2A  2A  00836             ;
      53  20  65  72  61  77  64  72  61  48  00845                 ;
2A  2A  20  65  6C  62  61  54  20  74  72  6F  70  70  75  0084F    .ASCII  \upport Table ******************!/\       ;
2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  0085E             ;
      2F  21  2A  0086D                 ;
6F  70  70  75  53  5F  65  72  61  77  64  72  61  48  26  00870 P.AHZ:  .ASCII  \&Hardware_Support [!1UL, !1UL]: . AC.!/\   ;
5D  4C  55  31  21  20  2C  4C  55  31  21  5B  20  74  72  0087F             ;
    2F  21  2E  43  41  21  2E  20  3A  0088E                 ;
```

```
2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  2A  14  00897 P.AIA:  .ASCII  <20>\********************\        ;
      2A  2A  2A  2A  2A  2A  008A6            ;
    2F  21  02  008AC P.AIB:  .ASCII  <2>\!/\          ;

        OUT_HS_HEADER=      P.AHY
        OUT_HS_TABLE=       P.AHZ
        T_NOTHING_THERE=    P.AIA
         .EXTRN  CRD$PRINT

         .PSECT  CODE,NOWRT,  SHR,  PIC,2

          001C 00000     .ENTRY  CRD$PRINT_HARDWARE_SUPPORT, Save R2,R3,R4    ; 1036
    00000000'  EF  9F 00002     PUSHAB  OUT_HS_HEADER           ; 1065
        01  DD 00008     PUSHL   #1              ;
        1A  DD 0000A     PUSHL   #26             ;
    00000000G  FF       03  FB 0000C     CALLS   #3, @CRD$PRINT          ;
        52  D4 00013     CLRL    HS_DIAG             ; 1067
        54        52       05  78 00015 1$:   ASHL    #5, HS_DIAG, R4          ; 1091
        53  D4 00019     CLRL    HS_ENTRY            ;
        50    6443  DE 0001B 2$:   MOVAL   (R4)[HS_ENTRY], R0          ;
        05       52  D1 0001F     CMPL    HS_DIAG, #5             ;
        05  18 00022     BGEQ    3$              ;
        08       53  D1 00024     CMPL    HS_ENTRY, #8            ;
        05  19 00027     BLSS    4$              ;
        50       01  CE 00029 3$:   MNEGL   #1, R0            ;
        08  11 0002C     BRB     5$              ;
        50 00000000'EF40  9E 0002E 4$:   MOVAB   CRD$AL_IDC_HDWR_SPRT_TABLE[R0], R0        ;
        50       60  D0 00036 5$:   MOVL    (R0), R0            ;
        60  95 00039     TSTB    (R0)            ;
        07  12 0003B     BNEQ    6$              ;
        50 00000000'  EF  9E 0003D     MOVAB   T_NOTHING_THERE, R0          ;
        50  DD 00044 6$:   PUSHL   R0              ;
        0C  BB 00046     PUSHR   #^M<R2,R3>          ;
    00000000'  EF  9F 00048     PUSHAB  OUT_HS_TABLE            ;
        01  DD 0004E     PUSHL   #1              ;
        1A  DD 00050     PUSHL   #26             ;
    00000000G  FF       06  FB 00052     CALLS   #6, @CRD$PRINT          ;
        BE        53       07  F3 00059     AOBLEQ  #7, HS_ENTRY, 2$         ; 1069
    00000000'  EF  9F 0005D     PUSHAB  P.AIB           ; 1094
        01  DD 00063     PUSHL   #1              ;
        1A  DD 00065     PUSHL   #26             ;
    00000000G  FF       03  FB 00067     CALLS   #3, @CRD$PRINT          ;
        A3        52       04  F3 0006E     AOBLEQ  #4, HS_DIAG, 1$          ; 1067
     04 00072     RET                 ; 1099

; Routine Size:  115 bytes,    Routine Base:  CODE + 09B5
```

```
;   1100  1 END     ! End of CRDHDWSUP module
;   1101  0 ELUDOM
```

```
;           PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          2223  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK           840  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          2600  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
$ASCIC     Macro    Lib02   155     156     157     158     160     161     162     163     169     170
              171     172     174     175     176     177     183     184     185     186
              188     189     190     191     197     198     199     200     202     203
              204     205     211     212     213     214     216     217     218     219
              230     231     232     233     235     236     237     238     244     245
              246     247     249     250     251     252     281     282     283     284
              286     287     288     289     295     296     297     298     300     301
              302     303     309     310     311     312     314     315     316     317
              323     324     325     326     328     329     330     331     346     347
              348     349     351     352     353     354     360     361     362     363
              365     366     367     368     374     375     376     377     379     380
              381     382     388     389     390     391     393     394     395     396
              402     403     404     405     407     408     409     410     422     423
              424     425     427     428     429     430     436     437     438     439
              441     442     443     444     450     451     452     453     455     456
              457     458     464     465     466     467     469     470     471     472
              486     487     488     489     491     492     493     494     500     501
              502     503     505     506     507     508     514     515     516     517
              519     520     521     522     528     529     530     531     533     534
              535     536     542     543     544     545     547     548     549     550
              556     557     558     559     560    1061    1062    1063    1094
$CRD_LITERALS   Macro    Lib03    91
$CRD_PRINT   Macro    Lib03  1065    1071    1094
$DS_TYPEDEF   Macro    Lib02  1065    1091    1094
$EQULST    Macro    Lib04    91    1065    1091    1094
$ER    Comptime   107
$MODULE    Bind       107
ASCIC_STRING   Bind      1091  1091.   1091a
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     91
AUTO$K_EXIT_CONTROL_C  Literal     91
AUTO$K_EXIT_DUMMY_2  Literal     91
AUTO$K_EXIT_DUMMY_3  Literal     91
AUTO$K_EXIT_ERRORS_COMPLETE Literal     91
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     91
AUTO$K_EXIT_INTERNAL_ERROR Literal     91
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     91
AUTO$K_EXIT_LAST_REASON  Literal     91     91
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     91
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     91
AUTO$K_EXIT_NOERRORS_PREP Literal     91
CRD$AL_IDC_HDWR_SPRT_TABLE Global     123 Lib03e    681     681=    688=    689=    690=    691=    692=    693=    694=
              695=    697=    698=    699=    700=    701=    702=    703=    704=    706=
              707=    708=    709=    710=    711=    712=    713=    715=    716=    717=
              718=    719=    720=    721=    722=    724=    725=    726=    727=    728=
              729=    730=    731=   1091    1091.
CRD$AL_LESI_HDWR_SPRT_TABLE Global     126 Lib03e    746     746=    753=    754=    755=    756=    757=    758=    759=
              760=    762=    763=    764=    765=    766=    767=    768=    769=
CRD$AL_UDA_0_HDWR_SPRT_TABLE Global     129 Lib03e    796     796=    803=    804=    805=    806=    807=    808=    809=
              810=    812=    813=    814=    815=    816=    817=    818=    819=    821=
              822=    823=    824=    825=    826=    827=    828=    830=    831=    832=
              833=    834=    835=    836=    837=
CRD$AL_UDA_1_HDWR_SPRT_TABLE Global     133 Lib03e    855     855=    862=    863=    864=    865=    866=    867=    868=
              869=    871=    872=    873=    874=    875=    876=    877=    878=    880=
              881=    882=    883=    884=    885=    886=    887=    889=    890=    891=
              892=    893=    894=    895=    896=    898=    899=    900=    901=    902=
              903=    904=    905=
CRD$AL_UDA_2_HDWR_SPRT_TABLE Global     137 Lib03e    921     921=    928=    929=    930=    931=    932=    933-    934-
              935=    937=    938=    939=    940=    941=    942=    943=    944=    946=
              947=    948=    949=    950=    951=    952=    953=    955=    956=    957=
              958=    959=    960=    961=    962=
CRD$AL_UDA_3_HDWR_SPRT_TABLE Global     141 Lib03e    980     980=    987=    988=    989=    990=    991=    992=    993=
              994=    996=    997=    998=    999=   1000=   1001=   1002=   1003=   1005=
             1006=   1007=   1008=   1009=   1010=   1011=   1012=   1014=   1015=   1016=
             1017=   1018=   1019=   1020=   1021=   1023=   1024=   1025=   1026-   1027=
             1028=   1029=   1030=
CRD$BUILD_SUPPORT_TABLE  GlobRout   563 Lib03e     84f
CRD$GA_HDWR_SPRT_TABLE  Global     117 Lib03e    117=
CRD$GB_BASE_SYSTEM  Global     111 Lib03e    111=
CRD$GT_DS_LOAD_COM  External Lib03
    GlobBind   556
CRD$GT_DS_SELECT_COM  External Lib03
    GlobBind   559
CRD$GT_DS_SET_EVENT_COM  External Lib03
    GlobBind   558
CRD$GT_DS_SET_FLAGS_COM  External Lib03
    GlobBind   557
CRD$GT_DS_START_COM  External Lib03
    GlobBind   560
CRD$GT_IDC_EVDLB  Bind       183   706a
CRD$GT_IDC_EVDLB_DEV_NAME Bind       186   710a
CRD$GT_IDC_EVDLB_PREP_ERR Bind       191   713a
CRD$GT_IDC_EVDLB_RUNTIME Bind       190   712a
CRD$GT_IDC_EVDLB_SET_EVNT Bind       185   708a
CRD$GT_IDC_EVDLB_SET_FLGS Bind       184   707a
CRD$GT_IDC_EVDLB_STRT_QUAL Bind       188   709a
CRD$GT_IDC_EVDLB_TST_STRT Bind       189   711a
```

Symbol	Type	Defined	Referenced ...
CRD$GT_IDC_EVDLC	Bind	197	715a
CRD$GT_IDC_EVDLC_DEV_NAME	Bind	200	719a
CRD$GT_IDC_EVDLC_PREP_ERR	Bind	205	722a
CRD$GT_IDC_EVDLC_RUNTIME	Bind	204	721a
CRD$GT_IDC_EVDLC_SET_EVNT	Bind	199	717a
CRD$GT_IDC_EVDLC_SET_FLGS	Bind	198	716a
CRD$GT_IDC_EVDLC_STRT_QUAL	Bind	202	718a
CRD$GT_IDC_EVDLC_TST_STRT	Bind	203	720a
CRD$GT_IDC_EVDLD	Bind	211	724a
CRD$GT_IDC_EVDLD_DEV_NAME	Bind	214	728a
CRD$GT_IDC_EVDLD_PREP_ERR	Bind	219	731a
CRD$GT_IDC_EVDLD_RUNTIME	Bind	218	730a
CRD$GT_IDC_EVDLD_SET_EVNT	Bind	213	726a
CRD$GT_IDC_EVDLD_SET_FLGS	Bind	212	725a
CRD$GT_IDC_EVDLD_STRT_QUAL	Bind	216	727a
CRD$GT_IDC_EVDLD_TST_STRT	Bind	217	729a
CRD$GT_IDC_EVRAD_0	Bind	155	688a
CRD$GT_IDC_EVRAD_0_DEV_NAME	Bind	158	692a
CRD$GT_IDC_EVRAD_0_PREP_ERR	Bind	163	695a
CRD$GT_IDC_EVRAD_0_RUNTIME	Bind	162	694a
CRD$GT_IDC_EVRAD_0_SET_EVNT	Bind	157	690a
CRD$GT_IDC_EVRAD_0_SET_FLGS	Bind	156	689a
CRD$GT_IDC_EVRAD_0_STRT_QUAL	Bind	160	691a
CRD$GT_IDC_EVRAD_0_TST_STRT	Bind	161	693a
CRD$GT_IDC_EVRAD_1	Bind	169	697a
CRD$GT_IDC_EVRAD_1_DEV_NAME	Bind	172	701a
CRD$GT_IDC_EVRAD_1_PREP_ERR	Bind	177	704a
CRD$GT_IDC_EVRAD_1_RUNTIME	Bind	176	703a
CRD$GT_IDC_EVRAD_1_SET_EVNT	Bind	171	699a
CRD$GT_IDC_EVRAD_1_SET_FLGS	Bind	170	698a
CRD$GT_IDC_EVRAD_1_STRT_QUAL	Bind	174	700a
CRD$GT_IDC_EVRAD_1_TST_STRT	Bind	175	702a
CRD$GT_LESI_EVRMB_0	Bind	230	753a
CRD$GT_LESI_EVRMB_0_DEV_NAME	Bind	233	757a
CRD$GT_LESI_EVRMB_0_PREP_ERR	Bind	238	760a
CRD$GT_LESI_EVRMB_0_RUNTIME	Bind	237	759a
CRD$GT_LESI_EVRMB_0_SET_EVNT	Bind	232	755a
CRD$GT_LESI_EVRMB_0_SET_FLGS	Bind	231	754a
CRD$GT_LESI_EVRMB_0_STRT_QUAL	Bind	235	756a
CRD$GT_LESI_EVRMB_0_TST_STRT	Bind	236	758a
CRD$GT_LESI_EVRMB_1	Bind	244	762a
CRD$GT_LESI_EVRMB_1_DEV_NAME	Bind	247	766a
CRD$GT_LESI_EVRMB_1_PREP_ERR	Bind	252	769a
CRD$GT_LESI_EVRMB_1_RUNTIME	Bind	251	768a
CRD$GT_LESI_EVRMB_1_SET_EVNT	Bind	246	764a
CRD$GT_LESI_EVRMB_1_SET_FLGS	Bind	245	763a
CRD$GT_LESI_EVRMB_1_STRT_QUAL	Bind	249	765a
CRD$GT_LESI_EVRMB_1_TST_STRT	Bind	250	767a
CRD$GT_UDA_0_EVDLB	Bind	295	812a
CRD$GT_UDA_0_EVDLB_DEV_NAME	Bind	298	816a
CRD$GT_UDA_0_EVDLB_PREP_ERR	Bind	303	819a
CRD$GT_UDA_0_EVDLB_RUNTIME	Bind	302	818a
CRD$GT_UDA_0_EVDLB_SET_EVNT	Bind	297	814a

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$GT_UDA_0_EVDLB_SET_FLGS Bind        296   813a
CRD$GT_UDA_0_EVDLB_STRT_QUAL Bind       300   815a
CRD$GT_UDA_0_EVDLB_TST_STRT Bind        301   817a
CRD$GT_UDA_0_EVDLC  Bind        309   821a
CRD$GT_UDA_0_EVDLC_DEV_NAME Bind        312   825a
CRD$GT_UDA_0_EVDLC_PREP_ERR Bind        317   828a
CRD$GT_UDA_0_EVDLC_RUNTIME Bind         316   827a
CRD$GT_UDA_0_EVDLC_SET_EVNT Bind        311   823a
CRD$GT_UDA_0_EVDLC_SET_FLGS Bind        310   822a
CRD$GT_UDA_0_EVDLC_STRT_QUAL Bind       314   824a
CRD$GT_UDA_0_EVDLC_TST_STRT Bind        315   826a
CRD$GT_UDA_0_EVDLD  Bind        323   830a
CRD$GT_UDA_0_EVDLD_DEV_NAME Bind        326   834a
CRD$GT_UDA_0_EVDLD_PREP_ERR Bind        331   837a
CRD$GT_UDA_0_EVDLD_RUNTIME Bind         330   836a
CRD$GT_UDA_0_EVDLD_SET_EVNT Bind        325   832a
CRD$GT_UDA_0_EVDLD_SET_FLGS Bind        324   831a
CRD$GT_UDA_0_EVDLD_STRT_QUAL Bind       328   833a
CRD$GT_UDA_0_EVDLD_TST_STRT Bind        329   835a
CRD$GT_UDA_0_EVRLA_0  Bind      281   803a
CRD$GT_UDA_0_EVRLA_0_DEV_NAME Bind      284   807a
CRD$GT_UDA_0_EVRLA_0_PREP_ERR Bind      289   810a
CRD$GT_UDA_0_EVRLA_0_RUNTIME Bind       288   809a
CRD$GT_UDA_0_EVRLA_0_SET_EVNT Bind      283   805a
CRD$GT_UDA_0_EVRLA_0_SET_FLGS Bind      282   804a
CRD$GT_UDA_0_EVRLA_0_STRT_QUAL Bind     286   806a
CRD$GT_UDA_0_EVRLA_0_TST_STRT Bind      287   808a
CRD$GT_UDA_1_EVDLB  Bind        374   880a
CRD$GT_UDA_1_EVDLB_DEV_NAME Bind        377   884a
CRD$GT_UDA_1_EVDLB_PREP_ERR Bind        382   887a
CRD$GT_UDA_1_EVDLB_RUNTIME Bind         381   886a
CRD$GT_UDA_1_EVDLB_SET_EVNT Bind        376   882a
CRD$GT_UDA_1_EVDLB_SET_FLGS Bind        375   881a
CRD$GT_UDA_1_EVDLB_STRT_QUAL Bind       379   883a
CRD$GT_UDA_1_EVDLB_TST_STRT Bind        380   885a
CRD$GT_UDA_1_EVDLC  Bind        388   889a
CRD$GT_UDA_1_EVDLC_DEV_NAME Bind        391   893a
CRD$GT_UDA_1_EVDLC_PREP_ERR Bind        396   896a
CRD$GT_UDA_1_EVDLC_RUNTIME Bind         395   895a
CRD$GT_UDA_1_EVDLC_SET_EVNT Bind        390   891a
CRD$GT_UDA_1_EVDLC_SET_FLGS Bind        389   890a
CRD$GT_UDA_1_EVDLC_STRT_QUAL Bind       393   892a
CRD$GT_UDA_1_EVDLC_TST_STRT Bind        394   894a
CRD$GT_UDA_1_EVDLD  Bind        402   898a
CRD$GT_UDA_1_EVDLD_DEV_NAME Bind        405   902a
CRD$GT_UDA_1_EVDLD_PREP_ERR Bind        410   905a
CRD$GT_UDA_1_EVDLD_RUNTIME Bind         409   904a
CRD$GT_UDA_1_EVDLD_SET_EVNT Bind        404   900a
CRD$GT_UDA_1_EVDLD_SET_FLGS Bind        403   899a
CRD$GT_UDA_1_EVDLD_STRT_QUAL Bind       407   901a
CRD$GT_UDA_1_EVDLD_TST_STRT Bind        408   903a
CRD$GT_UDA_1_EVRLA_0  Bind      346   862a
CRD$GT_UDA_1_EVRLA_0_DEV_NAME Bind      349   866a
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------- -------- ----- ---------------
CRD$GT_UDA_1_EVRLA_0_PREP_ERR Bind       354   869a
CRD$GT_UDA_1_EVRLA_0_RUNTIME Bind       353   868a
CRD$GT_UDA_1_EVRLA_0_SET_EVNT Bind       348   864a
CRD$GT_UDA_1_EVRLA_0_SET_FLGS Bind       347   863a
CRD$GT_UDA_1_EVRLA_0_STRT_QUAL Bind       351   865a
CRD$GT_UDA_1_EVRLA_0_TST_STRT Bind       352   867a
CRD$GT_UDA_1_EVRLA_1  Bind       360   871a
CRD$GT_UDA_1_EVRLA_1_DEV_NAME Bind       363   875a
CRD$GT_UDA_1_EVRLA_1_PREP_ERR Bind       368   878a
CRD$GT_UDA_1_EVRLA_1_RUNTIME Bind       367   877a
CRD$GT_UDA_1_EVRLA_1_SET_EVNT Bind       362   873a
CRD$GT_UDA_1_EVRLA_1_SET_FLGS Bind       361   872a
CRD$GT_UDA_1_EVRLA_1_STRT_QUAL Bind       365   874a
CRD$GT_UDA_1_EVRLA_1_TST_STRT Bind       366   876a
CRD$GT_UDA_2_EVDLB  Bind       436   937a
CRD$GT_UDA_2_EVDLB_DEV_NAME Bind       439   941a
CRD$GT_UDA_2_EVDLB_PREP_ERR Bind       444   944a
CRD$GT_UDA_2_EVDLB_RUNTIME Bind       443   943a
CRD$GT_UDA_2_EVDLB_SET_EVNT Bind       438   939a
CRD$GT_UDA_2_EVDLB_SET_FLGS Bind       437   938a
CRD$GT_UDA_2_EVDLB_STRT_QUAL Bind       441   940a
CRD$GT_UDA_2_EVDLB_TST_STRT Bind       442   942a
CRD$GT_UDA_2_EVDLC  Bind       450   946a
CRD$GT_UDA_2_EVDLC_DEV_NAME Bind       453   950a
CRD$GT_UDA_2_EVDLC_PREP_ERR Bind       458   953a
CRD$GT_UDA_2_EVDLC_RUNTIME Bind       457   952a
CRD$GT_UDA_2_EVDLC_SET_EVNT Bind       452   948a
CRD$GT_UDA_2_EVDLC_SET_FLGS Bind       451   947a
CRD$GT_UDA_2_EVDLC_STRT_QUAL Bind       455   949a
CRD$GT_UDA_2_EVDLC_TST_STRT Bind       456   951a
CRD$GT_UDA_2_EVDLD  Bind       464   955a
CRD$GT_UDA_2_EVDLD_DEV_NAME Bind       467   959a
CRD$GT_UDA_2_EVDLD_PREP_ERR Bind       472   962a
CRD$GT_UDA_2_EVDLD_RUNTIME Bind       471   961a
CRD$GT_UDA_2_EVDLD_SET_EVNT Bind       466   957a
CRD$GT_UDA_2_EVDLD_SET_FLGS Bind       465   956a
CRD$GT_UDA_2_EVDLD_STRT_QUAL Bind       469   958a
CRD$GT_UDA_2_EVDLD_TST_STRT Bind       470   960a
CRD$GT_UDA_2_EVRLA_0  Bind       422   928a
CRD$GT_UDA_2_EVRLA_0_DEV_NAME Bind       425   932a
CRD$GT_UDA_2_EVRLA_0_PREP_ERR Bind       430   935a
CRD$GT_UDA_2_EVRLA_0_RUNTIME Bind       429   934a
CRD$GT_UDA_2_EVRLA_0_SET_EVNT Bind       424   930a
CRD$GT_UDA_2_EVRLA_0_SET_FLGS Bind       423   929a
CRD$GT_UDA_2_EVRLA_0_STRT_QUAL Bind       427   931a
CRD$GT_UDA_2_EVRLA_0_TST_STRT Bind       428   933a
CRD$GT_UDA_3_EVDLB  Bind       514  1005a
CRD$GT_UDA_3_EVDLB_DEV_NAME Bind       517  1009a
CRD$GT_UDA_3_EVDLB_PREP_ERR Bind       522  1012a
CRD$GT_UDA_3_EVDLB_RUNTIME Bind       521  1011a
CRD$GT_UDA_3_EVDLB_SET_EVNT Bind       516  1007a
CRD$GT_UDA_3_EVDLB_SET_FLGS Bind       515  1006a
CRD$GT_UDA_3_EVDLB_STRT_QUAL Bind       519  1008a
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------------- ----- ------- ---------------
CRD$GT_UDA_3_EVDLB_TST_STRT Bind        520  1010a
CRD$GT_UDA_3_EVDLC  Bind        528  1014a
CRD$GT_UDA_3_EVDLC_DEV_NAME Bind        531  1018a
CRD$GT_UDA_3_EVDLC_PREP_ERR Bind        536  1021a
CRD$GT_UDA_3_EVDLC_RUNTIME Bind        535  1020a
CRD$GT_UDA_3_EVDLC_SET_EVNT Bind        530  1016a
CRD$GT_UDA_3_EVDLC_SET_FLGS Bind        529  1015a
CRD$GT_UDA_3_EVDLC_STRT_QUAL Bind        533  1017a
CRD$GT_UDA_3_EVDLC_TST_STRT Bind        534  1019a
CRD$GT_UDA_3_EVDLD  Bind        542  1023a
CRD$GT_UDA_3_EVDLD_DEV_NAME Bind        545  1027a
CRD$GT_UDA_3_EVDLD_PREP_ERR Bind        550  1030a
CRD$GT_UDA_3_EVDLD_RUNTIME Bind        549  1029a
CRD$GT_UDA_3_EVDLD_SET_EVNT Bind        544  1025a
CRD$GT_UDA_3_EVDLD_SET_FLGS Bind        543  1024a
CRD$GT_UDA_3_EVDLD_STRT_QUAL Bind        547  1026a
CRD$GT_UDA_3_EVDLD_TST_STRT Bind        548  1028a
CRD$GT_UDA_3_EVRLA_0  Bind        486   987a
CRD$GT_UDA_3_EVRLA_0_DEV_NAME Bind        489   991a
CRD$GT_UDA_3_EVRLA_0_PREP_ERR Bind        494   994a
CRD$GT_UDA_3_EVRLA_0_RUNTIME Bind        493   993a
CRD$GT_UDA_3_EVRLA_0_SET_EVNT Bind        488   989a
CRD$GT_UDA_3_EVRLA_0_SET_FLGS Bind        487   988a
CRD$GT_UDA_3_EVRLA_0_STRT_QUAL Bind        491   990a
CRD$GT_UDA_3_EVRLA_0_TST_STRT Bind        492   992a
CRD$GT_UDA_3_EVRLA_1  Bind        500   996a
CRD$GT_UDA_3_EVRLA_1_DEV_NAME Bind        503  1000a
CRD$GT_UDA_3_EVRLA_1_PREP_ERR Bind        508  1003a
CRD$GT_UDA_3_EVRLA_1_RUNTIME Bind        507  1002a
CRD$GT_UDA_3_EVRLA_1_SET_EVNT Bind        502   998a
CRD$GT_UDA_3_EVRLA_1_SET_FLGS Bind        501   997a
CRD$GT_UDA_3_EVRLA_1_STRT_QUAL Bind        505   999a
CRD$GT_UDA_3_EVRLA_1_TST_STRT Bind        506  1001a
CRD$K_ACTION_EQL_DO_NOTHING Literal      91
CRD$K_ACTION_RANGE_ERROR Literal      91
CRD$K_ADVANCE_DIAGNOSTIC Literal      91
CRD$K_ADVANCE_GENERAL_COM Literal      91
CRD$K_ADVANCE_STATE  Literal      91
CRD$K_ALLOCATION_ERROR  Literal      91
CRD$K_ASSIGN_PTABLE_PTRS Literal      91
CRD$K_AUTOSIZER_ERROR  Literal      91
CRD$K_AUTOSIZER_SET_FLAGS Literal      91
CRD$K_BAD_ACTION_NUMBER  Literal      91
CRD$K_BAD_TYPECODE_ERROR Literal      91
CRD$K_BASE_SYSTEM_IDC  Literal      91
CRD$K_BASE_SYSTEM_LESI  Literal      91
CRD$K_BASE_SYSTEM_NULL  Literal      91   111
CRD$K_BASE_SYSTEM_UDA_0  Literal      91
CRD$K_BASE_SYSTEM_UDA_1  Literal      91
CRD$K_BASE_SYSTEM_UDA_2  Literal      91
CRD$K_BASE_SYSTEM_UDA_3  Literal      91
CRD$K_CRD_INITIALIZATION Literal      91
CRD$K_CREATE_HST  Literal      91
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_DATA_LENGTH_ERROR  Literal     91
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     91
CRD$K_DDB_BLINK   Literal     91
CRD$K_DDB_FLINK   Literal     91
CRD$K_DDB_LAST_ENTRY  Literal     91
CRD$K_DDB_MEMORY_SIZE  Literal     91
CRD$K_DDB_MENU_DEVICE_NUMB Literal     91
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     91
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     91
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     91
CRD$K_DDB_PTABLE_ENTRY  Literal     91
CRD$K_DDB_STATUS  Literal     91
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     91
CRD$K_DEVICE_NAME  Unbound          636
    Literal     91   692     701     710     719     728     757     766     807     816     825
               834    866     875     884     893     902     932     941     950     959
               991   1000    1009    1018    1027
CRD$K_DIAGNOSTIC_ERROR  Literal     91
CRD$K_DIAGNOSTIC_NAME  Unbound          597
    Literal     91   688     697     706     715     724     753     762     803     812     821
               830    862     871     880     889     898     928     937     946     955
               987    996    1005    1014    1023
CRD$K_DONE_GENERAL_COMS  Literal     91
CRD$K_DO_NOTHING  Literal     91
CRD$K_DUMMY_10   Literal     91
CRD$K_DUMMY_11   Literal     91
CRD$K_DUMMY_12   Literal     91
CRD$K_DUMMY_13   Literal     91
CRD$K_DUMMY_3   Literal     91
CRD$K_DUMMY_4   Literal     91
CRD$K_DUMMY_5   Literal     91
CRD$K_DUMMY_6   Literal     91
CRD$K_DUMMY_7   Literal     91
CRD$K_DUMMY_8   Literal     91
CRD$K_DUMMY_9   Literal     91
CRD$K_EXIT_CRD   Literal     91
CRD$K_HOOK_POINT  Literal     91
CRD$K_HS_INTERNAL_ERROR  Literal     91
CRD$K_IDC_EVDLB   Literal     91   706     707     708     709     710     711     712     713
CRD$K_IDC_EVDLC   Literal     91   715     716     717     718     719     720     721     722
CRD$K_IDC_EVDLD   Literal     91   724     725     726     727     728     729     730     731
CRD$K_IDC_EVRAD_0  Literal     91   688     689     690     691     692     693     694     695
CRD$K_IDC_EVRAD_1  Literal     91   697     698     699     700     701     702     703     704
CRD$K_IDC_LAST_DIAGNOSTIC Literal     91   678
CRD$K_IDC_NUMB_DIAGNOSTICS Literal  Lib03   124    1067
CRD$K_INTERNAL_ERROR  Literal     91
CRD$K_LAST_ACTION_ROUTINE Literal     91
CRD$K_LAST_ENTRY  Literal     91   679     744     794     853     919     978
CRD$K_LAST_FUNCTION  Literal     91
CRD$K_LAST_HOOK_POINT  Literal     91
CRD$K_LAST_INTERNAL_ERROR Literal     91
CRD$K_LAST_TYPE   Literal     91
CRD$K_LESI_ENWCA  Literal     91
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_LESI_EVRMB_0  Literal      91   753     754     755     756     757     758     759     760
CRD$K_LESI_EVRMB_1  Literal      91   762     763     764     765     766     767     768     769
CRD$K_LESI_LAST_DIAGNOSTIC Literal     91   743
CRD$K_LESI_NUMB_DIAGNOSTICS Literal  Lib03   127
CRD$K_LEVEL_4_PASSED  Literal      91
CRD$K_LOAD_AUTOSIZER  Literal      91
CRD$K_LOAD_ERROR  Literal      91
CRD$K_LOAD_GENERAL_COM  Literal      91
CRD$K_LOAD_LOAD_COM  Literal      91
CRD$K_LOAD_SELECT_COM  Literal      91
CRD$K_LOAD_SET_EVENT_COM Literal      91
CRD$K_LOAD_SET_FLAGS_COM Literal      91
CRD$K_LOAD_START_COM  Literal      91
CRD$K_LOAD_SUP_FILE  Literal      91
CRD$K_MM_INTERNAL_ERROR  Literal      91
CRD$K_NEW_LINE   Literal      91
CRD$K_NUMB_SUPPORT_ENTRIES Literal  Lib03   124     127     130     134     138     142    1069
CRD$K_PREPARATION_ERROR  Literal      91
CRD$K_PREPARATION_ERROR_TEXT Unbound          656
    Literal      91   695     704     713     722     731     760     769     810     819     828
                837     869     878     887     896     905     935     944     953     962
                994    1003    1012    1021    1030
CRD$K_RUNTIME   Unbound          646
    Literal      91   694     703     712     721     730     759     768     809     818     827
                836     868     877     886     895     904     934     943     952     961
                993    1002    1011    1020    1029
CRD$K_SAME_LINE   Literal      91
CRD$K_SET_EVENT_ARGS  Unbound          617
    Literal      91   690     699     708     717     726     755     764     805     814     823
                832     864     873     882     891     900     930     939     948     957
                989     998    1007    1016    1025
CRD$K_SET_FLAGS_ARGS  Unbound          607
    Literal      91   689     698     707     716     725     754     763     804     813     822
                831     863     872     881     890     899     929     938     947     956
                988     997    1006    1015    1024
CRD$K_SET_UP_DIAGNOSTIC  Literal      91
CRD$K_START   Literal      91
CRD$K_START_AUTOSIZER  Literal      91
CRD$K_START_ERROR  Literal      91
CRD$K_START_QUALIFIERS  Unbound          627
    Literal      91   691     700     709     718     727     756     765     806     815     824
                833     865     874     883     892     901     931     940     949     958
                990     999    1008    1017    1026
CRD$K_STAR_LINE   Literal      91
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal      91
CRD$K_STATE_ADVANCE_GENERAL_COM Literal      91
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal      91
CRD$K_STATE_AUTOSIZER_ERROR Literal      91
CRD$K_STATE_AUTOSIZER_RUNNING Literal      91
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal      91
CRD$K_STATE_DIAGNOSTIC_ERROR Literal      91
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal      91
CRD$K_STATE_DONE_DIAGNOSTICS Literal      91
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_DONE_GENERAL_COMS Literal     91
CRD$K_STATE_DUMMY_1  Literal     91
CRD$K_STATE_DUMMY_10  Literal     91
CRD$K_STATE_DUMMY_12  Literal     91
CRD$K_STATE_DUMMY_13  Literal     91
CRD$K_STATE_DUMMY_2  Literal     91
CRD$K_STATE_DUMMY_3  Literal     91
CRD$K_STATE_DUMMY_4  Literal     91
CRD$K_STATE_DUMMY_6  Literal     91
CRD$K_STATE_DUMMY_7  Literal     91
CRD$K_STATE_DUMMY_8  Literal     91
CRD$K_STATE_EXIT_CRD  Literal     91
CRD$K_STATE_INTERNAL_ERROR Literal     91
CRD$K_STATE_LAST_STATE  Literal     91
CRD$K_STATE_LEVEL_4_PASSED Literal     91
CRD$K_STATE_LOAD_AUTOSIZER Literal     91
CRD$K_STATE_LOAD_ERROR  Literal     91
CRD$K_STATE_LOAD_GENERAL_COM Literal     91
CRD$K_STATE_LOAD_LOAD_COM Literal     91
CRD$K_STATE_LOAD_SELECT_COM Literal     91
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     91
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     91
CRD$K_STATE_LOAD_START_COM Literal     91
CRD$K_STATE_PREPARATION_ERROR Literal     91
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     91
CRD$K_STATE_START  Literal     91
CRD$K_STATE_START_AUTOSIZER Literal     91
CRD$K_STATE_START_ERROR  Literal     91
CRD$K_TEST_START_TEXT  Unbound          666
    Literal     91   693     702     711     720     729     758     767     808     817     826
              835     867     876     885     894     903     933     942     951     960
              992    1001    1010    1019    1028
CRD$K_TQ_INTERNAL_ERROR  Literal     91
CRD$K_TYPECODE   Literal     91
CRD$K_UDA_0_EVDLB  Literal     91   812     813     814     815     816     817     818     819
CRD$K_UDA_0_EVDLC  Literal     91   821     822     823     824     825     826     827     828
CRD$K_UDA_0_EVDLD  Literal     91   830     831     832     833     834     835     836     837
CRD$K_UDA_0_EVRLA_0  Literal     91   803     804     805     806     807     808     809     810
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     91   793
CRD$K_UDA_0_NUMB_DIAGNOSTICS Literal  Lib03   130
CRD$K_UDA_1_EVDLB  Literal     91   880     881     882     883     884     885     886     887
CRD$K_UDA_1_EVDLC  Literal     91   889     890     891     892     893     894     895     896
CRD$K_UDA_1_EVDLD  Literal     91   898     899     900     901     902     903     904     905
CRD$K_UDA_1_EVRLA_0  Literal     91   862     863     864     865     866     867     868     869
CRD$K_UDA_1_EVRLA_1  Literal     91   871     872     873     874     875     876     877     878
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     91   852
CRD$K_UDA_1_NUMB_DIAGNOSTICS Literal  Lib03   134
CRD$K_UDA_2_EVDLB  Literal     91   937     938     939     940     941     942     943     944
CRD$K_UDA_2_EVDLC  Literal     91   946     947     948     949     950     951     952     953
CRD$K_UDA_2_EVDLD  Literal     91   955     956     957     958     959     960     961     962
CRD$K_UDA_2_EVRLA_0  Literal     91   928     929     930     931     932     933     934     935
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     91   918
CRD$K_UDA_2_NUMB_DIAGNOSTICS Literal  Lib03   138
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
CRD$K_UDA_3_EVDLB  Literal     91  1005    1006    1007    1008    1009    1010    1011    1012
CRD$K_UDA_3_EVDLC  Literal     91  1014    1015    1016    1017    1018    1019    1020    1021
CRD$K_UDA_3_EVDLD  Literal     91  1023    1024    1025    1026    1027    1028    1029    1030
CRD$K_UDA_3_EVRLA_0  Literal     91   987     988     989     990     991     992     993     994
CRD$K_UDA_3_EVRLA_1  Literal     91   996     997     998     999    1000    1001    1002    1003
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     91   977
CRD$K_UDA_3_NUMB_DIAGNOSTICS Literal  Lib03   142
CRD$PRINT   External Lib03  1065ac  1091ac  1094ac
CRD$PRINT_HARDWARE_SUPPORT GlobRout  1036 Lib03e     87f
DIAG    MacrForm   592   597     598
    MacrForm   602   607     608
    MacrForm   612   617     618
    MacrForm   622   627     628
    MacrForm   631   636     637
    MacrForm   641   646     647
    MacrForm   651   656     657
    MacrForm   661   666     667
    Local      678   681.
    Local      743   746.
    Local      793   796.
    Local      852   855.
    Local      918   921.
    Local      977   980.
DS$K_PRINTB   Literal   1065
    Literal   1091
    Literal   1094
DS$K_PRINTF   Literal   1065  1065
    Literal   1091  1091
    Literal   1094  1094
DS$K_PRINTI   Literal   1065
    Literal   1091
    Literal   1094
DS$K_PRINTX   Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ABORT_PROGRAM  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ABORT_TEST  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_COMMAND_ERR  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_COMMAND_OUT  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_CRD_AUTOTEST  Literal   1065  1065
    Literal   1091  1091
    Literal   1094  1094
DS$K_TYPE_DS_PROMPT  Literal   1065
    Literal   1091
    Literal   1094
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DS$K_TYPE_DS_START  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ERRDEV  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ERRHARD  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ERROR_BODY  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ERROR_END  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ERRPREP  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ERRSOFT  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ERRSUP  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ERRSYS  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_ERR_HALT  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_EXCEPTION  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_EXCEPTION_HEAD Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_FIRST_PASS  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_GENERAL  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_GENERAL_ERROR  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_NO_TESTS  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_PARAM_ERROR  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_PROGRAM_END  Literal   1065
    Literal   1091
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal   1094
DS$K_TYPE_PROGRAM_INFO  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_PROGRAM_START  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_QIO_INVADP  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_QIO_NODRIVER  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_QIO_WRONGVER  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_SCRIPT_ECHO  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_SCRIPT_PNF  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_SCRIPT_PROMPT  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_SCRIPT_SKIP  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_SEQUENCE_ERROR Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_START_ERR  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_START_LIST  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_SUMMARY  Literal   1065
    Literal   1091
    Literal   1094
DS$K_TYPE_USER_PROMPT  Literal   1065
    Literal   1091
    Literal   1094
ENTRY    Local      679   681.
    Local       744   746.
    Local       794   796.
    Local       853   855.
    Local       919   921.
    Local       978   980.
GET1ST_    Macro    Lib04    91    1065    1091    1094
GET2ND_    Unbound            91    1065    1091    1094
HARDWARE_SUPPORT_STRUCTURE Structur Lib03   124     127     130     134     138     142
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal     91
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
HS$K_ACTION_ADVANCE_GENERAL_COM Literal     91
HS$K_ACTION_ADVANCE_STATE Literal     91
HS$K_ACTION_CHANGE_TABLE Literal     91
HS$K_ACTION_DIAGNOSTIC_ERROR Literal     91
HS$K_ACTION_DONE_GENERAL_COMS Literal     91
HS$K_ACTION_DO_NOTHING  Literal     91
HS$K_ACTION_DUMMY_3  Literal     91
HS$K_ACTION_DUMMY_4  Literal     91
HS$K_ACTION_DUMMY_5  Literal     91
HS$K_ACTION_DUMMY_6  Literal     91
HS$K_ACTION_INIT_HS  Literal     91
HS$K_ACTION_INTERNAL_ERROR Literal     91
HS$K_ACTION_LAST_ACTION  Literal     91
HS$K_ACTION_LOAD_ERROR  Literal     91
HS$K_ACTION_LOAD_GENERAL_COM Literal     91
HS$K_ACTION_LOAD_LOAD_COM Literal     91
HS$K_ACTION_LOAD_SELECT_COM Literal     91
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     91
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     91
HS$K_ACTION_LOAD_START_COM Literal     91
HS$K_ACTION_PREPARATION_ERROR Literal     91
HS$K_ACTION_START_ERROR  Literal     91
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     91
HS$K_STATE_ADVANCE_GENERAL_COM Literal     91
HS$K_STATE_CHANGE_TABLE  Literal     91
HS$K_STATE_DIAGNOSTIC_ERROR Literal     91
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     91
HS$K_STATE_DONE_GENERAL_COMS Literal     91
HS$K_STATE_DUMMY_1  Literal     91
HS$K_STATE_DUMMY_2  Literal     91
HS$K_STATE_DUMMY_3  Literal     91
HS$K_STATE_DUMMY_4  Literal     91
HS$K_STATE_DUMMY_6  Literal     91
HS$K_STATE_INIT_HS  Literal     91
HS$K_STATE_INTERNAL_ERROR Literal     91
HS$K_STATE_LAST_STATE  Literal     91
HS$K_STATE_LOAD_ERROR  Literal     91
HS$K_STATE_LOAD_GENERAL_COM Literal     91
HS$K_STATE_LOAD_LOAD_COM Literal     91
HS$K_STATE_LOAD_SELECT_COM Literal     91
HS$K_STATE_LOAD_SET_EVENT_COM Literal     91
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     91
HS$K_STATE_LOAD_START_COM Literal     91
HS$K_STATE_PREPARATION_ERROR Literal     91
HS$K_STATE_START_ERROR  Literal     91
HS_DIAG    Local     1067  1091.
HS_ENTRY   Local     1069  1091.
INSERT_DEV_NAME   Macro      631   692     701     710     719     728     757     766     807     816     825
              834     866     875     884     893     902     932     941     950     959
              991    1000    1009    1018    1027
INSERT_DIAGNOSTIC_NAME  Macro      592   688     697     706     715     724     753     762     803     812     821
              830     862     871     880     889     898     928     937     946     955
              987     996    1005    1014    1023
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
INSERT_PREP_ERR    Macro      651   695     704     713     722     731     760     769     810     819     828
                  837     869     878     887     896     905     935     944     953     962
                  994    1003    1012    1021    1030
INSERT_RUNTIME     Macro      641   694     703     712     721     730     759     768     809     818     827
                  836     868     877     886     895     904     934     943     952     961
                  993    1002    1011    1020    1029
INSERT_SET_EVNT    Macro      612   690     699     708     717     726     755     764     805     814     823
                  832     864     873     882     891     900     930     939     948     957
                  989     998    1007    1016    1025
INSERT_SET_FLGS    Macro      602   689     698     707     716     725     754     763     804     813     822
                  831     863     872     881     890     899     929     938     947     956
                  988     997    1006    1015    1024
INSERT_STRT_QUAL   Macro      622   691     700     709     718     727     756     765     806     815     824
                  833     865     874     883     892     901     931     940     949     958
                  990     999    1008    1017    1026
INSERT_TST_STRT    Macro      661   693     702     711     720     729     758     767     808     817     826
                  835     867     876     885     894     903     933     942     951     960
                  992    1001    1010    1019    1028
MEN$K_HS_STATE_TABLE  Literal      91
MEN$K_MM_STATE_TABLE  Literal      91
MEN$K_TQ_STATE_TABLE  Literal      91
MENU$K_EXIT_AUTOSIZER_ERROR Literal      91
MENU$K_EXIT_INTERNAL_ERROR Literal      91
MENU$K_EXIT_LAST_REASON  Literal      91
MENU$K_EXIT_OPERATOR_ABORT Literal      91
MENU$K_EXIT_OPERATOR_STOP Literal      91
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal      91
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal      91
MM$K_ACTION_ADVANCE_STATE Literal      91
MM$K_ACTION_AUTOSIZER_ERROR Literal      91
MM$K_ACTION_BUILD_DDBS  Literal      91
MM$K_ACTION_BUILD_HSTS  Literal      91
MM$K_ACTION_CHANGE_TABLE Literal      91
MM$K_ACTION_DONE_INITS  Literal      91
MM$K_ACTION_DO_NOTHING  Literal      91
MM$K_ACTION_DUMMY_3  Literal      91
MM$K_ACTION_DUMMY_4  Literal      91
MM$K_ACTION_DUMMY_5  Literal      91
MM$K_ACTION_DUMMY_6  Literal      91
MM$K_ACTION_DUMMY_7  Literal      91
MM$K_ACTION_DUMMY_8  Literal      91
MM$K_ACTION_INTERNAL_ERROR Literal      91
MM$K_ACTION_LAST_ACTION  Literal      91
MM$K_ACTION_LOAD_AUTOSIZER Literal      91
MM$K_ACTION_MENU_PROMPT  Literal      91
MM$K_ACTION_PRINT_MENU  Literal      91
MM$K_ACTION_PRINT_MENU_HEADING Literal      91
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal      91
MM$K_ACTION_START_AUTOSIZER Literal      91
MM$K_STATE_AUTOSIZER_ERROR Literal      91
MM$K_STATE_AUTOSIZER_RUNNING Literal      91
MM$K_STATE_BUILD_DDBS  Literal      91
MM$K_STATE_BUILD_HSTS  Literal      91
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_STATE_CHANGE_TABLE  Literal     91
MM$K_STATE_DONE_INITS  Literal     91
MM$K_STATE_DUMMY_1  Literal     91
MM$K_STATE_DUMMY_2  Literal     91
MM$K_STATE_DUMMY_3  Literal     91
MM$K_STATE_DUMMY_5  Literal     91
MM$K_STATE_DUMMY_7  Literal     91
MM$K_STATE_DUMMY_8  Literal     91
MM$K_STATE_INTERNAL_ERROR Literal     91
MM$K_STATE_LAST_STATE  Literal     91
MM$K_STATE_LOAD_AUTOSIZER Literal     91
MM$K_STATE_MENU_PROMPT  Literal     91
MM$K_STATE_PRINT_MENU  Literal     91
MM$K_STATE_PRINT_MENU_HEADING Literal     91
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     91
MM$K_STATE_START  Literal     91
MM$K_STATE_START_AUTOSIZER Literal     91
MODNAM    Macro    Lib01   107
NUL2ND    Macro    Lib04    91    1065    1091    1094
OUT_HS_HEADER   Bind      1061  1065a
OUT_HS_TABLE   Bind      1062  1091a
SYSTEM    MacrForm   592   597     598
    MacrForm   602   607     608
    MacrForm   612   617     618
    MacrForm   622   627     628
    MacrForm   631   636     637
    MacrForm   641   646     647
    MacrForm   651   656     657
    MacrForm   661   666     667
TQ$K_ACTION_ADVANCE_STATE Literal     91
TQ$K_ACTION_CHANGE_TABLE Literal     91
TQ$K_ACTION_DONE_TEST_QUEUE Literal     91
TQ$K_ACTION_DO_NOTHING  Literal     91
TQ$K_ACTION_DUMMY_3  Literal     91
TQ$K_ACTION_DUMMY_4  Literal     91
TQ$K_ACTION_DUMMY_5  Literal     91
TQ$K_ACTION_GET_DDB  Literal     91
TQ$K_ACTION_INIT_TQ  Literal     91
TQ$K_ACTION_INTERNAL_ERROR Literal     91
TQ$K_ACTION_LAST_ACTION  Literal     91
TQ$K_STATE_CHANGE_TABLE  Literal     91
TQ$K_STATE_DONE_TEST_QUEUE Literal     91
TQ$K_STATE_DUMMY_1  Literal     91
TQ$K_STATE_DUMMY_2  Literal     91
TQ$K_STATE_DUMMY_3  Literal     91
TQ$K_STATE_DUMMY_4  Literal     91
TQ$K_STATE_DUMMY_5  Literal     91
TQ$K_STATE_GET_DDB  Literal     91
TQ$K_STATE_INIT_TQ  Literal     91
TQ$K_STATE_INTERNAL_ERROR Literal     91
TQ$K_STATE_LAST_STATE  Literal     91
T_NOTHING_THERE   Bind      1063  1091a
```

```
  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDHDWSUP.B32;1
    3  Module CRDHDWSUP
   71  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   74  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   77  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   80  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
 1101  Eludom CRDHDWSUP


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     a Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;          -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0        46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923          2          0        94          00:00.1
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486         26          5        62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3          0      1000          00:04.2




;     COMMAND QUALIFIERS

; BLISS CRDHDWSUP.B32/LIST=CRDHDWSUP.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDHDWSUP.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  2600 code + 3063 data bytes
; Run Time:    01:02.4
; Elapsed Time:    01:53.5
; Lines/CPU Min:    1057
; Lexemes/CPU-Min: 58581
```

```
; Memory Used: 613 pages
; Compilation Complete
```

```
;    0001  0 %TITLE '*** CRDHSACT Module'
;    0002  0 MODULE CRDHSACT (    ! Contains action routines for Hardware Support
;    0003  0   IDENT = '01-01'
;    0004  0   ) =
;    0005  0 !++
;    0006  0 ! COPYRIGHT (c) 1980, 1981
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! This module contains action routines for the Hardware Support State Table.
;    0032  0 !
;    0033  0 ! Author:
;    0034  0 !
;    0035  0 ! Jack Stansbury
;    0036  0 !
;    0037  0 ! Creation Date:
;    0038  0 !
;    0039  0 ! July 30, 1982
;    0040  0 !
;    0041  0 ! Version:
;    0042  0 !
;    0043  0 ! 6.9
;    0044  0 !
;    0045  0 ! Modified By:
;    0046  0 !
;    0047  0 ! 01 Jack Stansbury, March 3, 1983, Version 1.2
;    0048  0 !  Check to make sure that the user passed the address of the error text when printing
;    0049  0 !  it out in the preparation error text routine.
;    0050  0 !-
```

```
;    0051  0 %SBTTL 'Libraries'
;    0052  1 BEGIN     ! Begin the CRDHSACT module
;    0053  1
;    0054  1 LIBRARY
;    0055  1  '$DS';    ! Use Ds.L32 first
;    0056  1
;    0057  1 LIBRARY
;    0058  1  '$Diag';   ! Use Diag.L32 second
;    0059  1
;    0060  1 LIBRARY
;    0061  1  '$CRD';    ! Use CRD.L32 third
;    0062  1
;    0063  1 LIBRARY
;    0064  1  'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0065  1 %SBTTL 'Table of Contents'
;    0066  1
;    0067  1 FORWARD ROUTINE
;    0068  1  HS$Init_HS    : ADDRESSING_MODE (LONG_RELATIVE),
;    0069  1  HS$Advance_Diagnostic  : ADDRESSING_MODE (LONG_RELATIVE),
;    0070  1
;    0071  1  HS$Load_General_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0072  1  HS$Advance_General_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0073  1  HS$Done_General_Coms  : ADDRESSING_MODE (LONG_RELATIVE),
;    0074  1
;    0075  1  HS$Load_Load_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0076  1  HS$Load_Set_Flags_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0077  1  HS$Load_Set_Event_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0078  1  HS$Load_Select_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0079  1  HS$Load_Start_Com  : ADDRESSING_MODE (LONG_RELATIVE),
;    0080  1
;    0081  1  HS$Change_Table    : ADDRESSING_MODE (LONG_RELATIVE),
;    0082  1
;    0083  1  HS$Load_Error    : ADDRESSING_MODE (LONG_RELATIVE),
;    0084  1  HS$Start_Error    : ADDRESSING_MODE (LONG_RELATIVE),
;    0085  1  HS$Diagnostic_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    0086  1  HS$Preparation_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    0087  1  HS$Internal_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    0088  1
;    0089  1  CRD$Print_DS_Command  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    0090  1 %SBTTL 'Included Macros and Literals'
;    0091  1
;    0092  1  $DS_DSADef;    ! Define the DSA flags
;    0093  1  $CRD_Literals;    ! Define the CRD literals
```

```
;    0094  1 %SBTTL 'Psect Definitions'
;    0095  1
;    0096  1 PSECT
;    0097  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0098  1
;    0099  1 PSECT
;    0100  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0101  1
;    0102  1 PSECT
;    0103  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0104  1
;    0105  1 PSECT
;    0106  1     GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG RELATIVE));
```

```
;    0107  1 %SBTTL 'Module-Global Static Storage'
;    0108  1
;    0109  1  ModNam ('CRDHSACT');  ! Module name for ErrSup macro
```

```
;    0110  1 %SBTTL 'HS$Init_HS Routine'
;    0111  1
;    0112  1 GLOBAL ROUTINE HS$Init_HS =
;    0113  1
;    0114  1 !++
;    0115  1 ! Functional Description:
;    0116  1 !
;    0117  1 ! Describe function of routine
;    0118  1 !
;    0119  1 ! Formal Input Parameters:
;    0120  1 !
;    0121  1 ! None
;    0122  1 !
;    0123  1 ! Formal Output Parameters:
;    0124  1 !
;    0125  1 ! None
;    0126  1 !
;    0127  1 ! Side Effects:
;    0128  1 !
;    0129  1 ! None
;    0130  1 !
;    0131  1 ! Completion Codes:
;    0132  1 !
;    0133  1 ! None
;    0134  1 !--
```

```
;    0135  2 BEGIN     ! Routine HS$Init_HS
;    0136  2  MAP
;    0137  2   CRD$GA_HS_Current_DDB_Ptr : Device_Data_Block_Structure;
;    0138  2
;    0139  2  !+
;    0140  2  ! Initialize these two HS variables.
;    0141  2  !-
;    0142  2
;    0143  2  CRD$GA_HS_Current_HSB_Ptr  = 0;
;    0144  2  CRD$GL_HS_Current_HSB_Numb = 0;
;    0145  2
;    0146  2  !+
;    0147  2  ! Initialize all of the status bits for this DDB.
;    0148  2  !-
;    0149  2
;    0150  2  CRD$Init_DDB_Status (.CRD$GA_HS_Current_DDB_Ptr);
;    0151  2
;    0152  3  RETURN (CRD$K_Repeat_Turing)
;    0153  1 END;     ! Routine HS$Init_DS


            .TITLE  CRDHSACT *** CRDHSACT Module
            .IDENT  \01-01\

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

   54   43   41   53   48   44   52   43   08   00000 P.AAA:  .ASCII  <8>\CRDHSACT\          ;

          DSA$AL_APTMAIL=     65024
          DSA$GL_FLAGS=       65024
          DSA$GL_APTCOM=      65028
          DSA$GL_PASSES=      65032
          DSA$GL_UNITS=       65036
          DSA$GL_SECTNO=      65040
          DSA$GL_SID=        65044
          DSA$GL_MSGTYP=      65088
          DSA$GL_ERRNO=       65092
          DSA$GL_EVENT=       65096
          DSA$GL_SUBTNO=      65100
          DSA$GL_TESTNO=      65104
          DSA$GL_PASSNO=      65108
          DSA$GL_DEVLEN=      65112
          DSA$GL_DEVNAM=      65116
          DSA$GL_MSGPTR=      65128
          $MODULE=        P.AAA
            .EXTRN  HS$INIT_HS, HS$ADVANCE_DIAGNOSTIC
            .EXTRN  HS$LOAD_GENERAL_COM
            .EXTRN  HS$ADVANCE_GENERAL_COM
            .EXTRN  HS$DONE_GENERAL_COMS
            .EXTRN  HS$LOAD_LOAD_COM
            .EXTRN  HS$LOAD_SET_FLAGS_COM
            .EXTRN  HS$LOAD_SET_EVENT_COM
            .EXTRN  HS$LOAD_SELECT_COM
            .EXTRN  HS$LOAD_START_COM
            .EXTRN  HS$CHANGE_TABLE
```

```
        .EXTRN  HS$LOAD_ERROR, HS$START_ERROR
        .EXTRN  HS$DIAGNOSTIC_ERROR
        .EXTRN  HS$PREPARATION_ERROR
        .EXTRN  HS$INTERNAL_ERROR
        .EXTRN  CRD$PRINT_DS_COMMAND
        .EXTRN  CRD$GA_HS_CURRENT_DDB_PTR
        .EXTRN  CRD$GA_HS_CURRENT_HSB_PTR
        .EXTRN  CRD$GL_HS_CURRENT_HSB_NUMB
        .EXTRN  CRD$INIT_DDB_STATUS

        .PSECT  CODE,NOWRT,  SHR,  PIC,2

          0000 00000     .ENTRY  HS$INIT_HS, Save nothing        ; 0112
    00000000G  EF   D4 00002     CLRL    CRD$GA_HS_CURRENT_HSB_PTR        ; 0143
    00000000G  EF   D4 00008     CLRL    CRD$GL_HS_CURRENT_HSB_NUMB        ; 0144
    00000000G  EF   DD 0000E     PUSHL   CRD$GA_HS_CURRENT_DDB_PTR        ; 0150
    00000000G  EF       01  FB 00014     CALLS   #1, CRD$INIT_DDB_STATUS        ;
        50       02  D0 0001B     MOVL    #2, R0          ; 0152
     04 0001E     RET                 ; 0153

; Routine Size:  31 bytes,    Routine Base:  CODE + 0000
```

```
;    0154  1 %SBTTL 'HS$Advance_Diagnostic Routine'
;    0155  1
;    0156  1 GLOBAL ROUTINE HS$Advance_Diagnostic =
;    0157  1
;    0158  1 !++
;    0159  1 ! Functional Description:
;    0160  1 !
;    0161  1 ! Describe function of routine
;    0162  1 !
;    0163  1 ! Formal Input Parameters:
;    0164  1 !
;    0165  1 ! None
;    0166  1 !
;    0167  1 ! Formal Output Parameters:
;    0168  1 !
;    0169  1 ! None
;    0170  1 !
;    0171  1 ! Side Effects:
;    0172  1 !
;    0173  1 ! None
;    0174  1 !
;    0175  1 ! Completion Codes:
;    0176  1 !
;    0177  1 ! None
;    0178  1 ! -
```

```
;    0179  2 BEGIN     ! Routine HS$Advance_Diagnostic
;    0180  2  MAP
;    0181  2   CRD$GA_HS_Current_DDB_Ptr : REF Device_Data_Block_Structure;
;    0182  2
;    0183  2  IF (.CRD$GL_HS_Current_HSB_Numb EQL 0) THEN
;    0184  3   BEGIN
;    0185  3   !+
;    0186  3   ! This is the first 'Advance_Diagnostic' we have done for this set of HSB's. Get a
;    0187  3   ! pointer to the first HSB from the DDB.
;    0188  3   !-
;    0189  3
;    0190  3   CRD$GA_HS_Current_HSB_Ptr = .CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Menu_Support_Entry];
;    0191  3   END
;    0192  2  ELSE
;    0193  3   BEGIN
;    0194  3   !+
;    0195  3   ! This is not the first HSB in this set of HSB's. Increment the HSB_Ptr to the next HSB.
;    0196  3   ! Increment by (the number of longwords per HSB) * (4 bytes)
;    0197  3   !-
;    0198  3
;    0199  3   CRD$GA_HS_Current_HSB_Ptr = .CRD$GA_HS_Current_HSB_Ptr + CRD$K_Numb_Support_Entries * 4;
;    0200  3
;    0201  3   !+
;    0202  3   ! Check to see if we should print  PASS  or  *FAILED* . If so, go ahead and print it.
;    0203  3   ! Set the appropriate status bits.
;    0204  3   !-
;    0205  3
;    0206  3   IF (NOT .CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed>) THEN
;    0207  3    $CRD_Print (CRD$GT_Pass);
;    0208  3
;    0209  3   CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Device_Bad> =
;    0210  3    .CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Device_Bad> OR
;    0211  3    .CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Diagnostic_Error>;
;    0212  2   END;
;    0213  2
;    0214  2  IF (.CRD$GL_HS_Current_HSB_Numb EQL .CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Numb_Support_Tables]) THEN
;    0215  3   BEGIN
;    0216  3   !+
;    0217  3   ! We have finished doing all the hardware support blocks for this DDB. Change the state
;    0218  3   ! table to the Test Queue State Table.
;    0219  3   !-
;    0220  3   CRD$GL_HS_Current_State = HS$K_State_Change_Table;
;    0221  3   END
;    0222  2  ELSE
;    0223  3   BEGIN
;    0224  3   !+
;    0225  3   ! We still have at least one more Hardware Support Block to run. Increment the running
;    0226  3   ! counter of HSB's done so far.
;    0227  3   !-
;    0228  3
;    0229  3   CRD$GL_HS_Current_HSB_Numb = .CRD$GL_HS_Current_HSB_Numb + 1;
;    0230  3
;    0231  3   !+
;    0232  3   ! Print the testing started message.
;    0233  3   !-
```

```
;    0234  3
;    0235  3    MEN$FncTst_TestStrt_Text (.CRD$GA_HS_Current_DDB_Ptr, .CRD$GA_HS_Current_HSB_Ptr);
;    0236  2    END;
;    0237  2
;    0238  2    !+
;    0239  2    ! Before starting the test, assume the diagnostic will not detect any errors, and that no
;    0240  2    ! messages have been printed for it so far.
;    0241  2    !-
;    0242  2
;    0243  2    CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Diagnostic_Error> = False;
;    0244  2    CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed>  = False;
;    0245  2
;    0246  2    IF (.CRD$GB_CRD_Debug) THEN
;    0247  3    BEGIN
; P  0248  3    $CRD_Print ($ASCIC ('*** Advance_Diagnostic: Status bits for !XL DDB: !XL!/'),
;    0249  3      .CRD$GA_HS_Current_DDB_Ptr, .CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status]);
;    0250  2    END;
;    0251  2
;    0252  3    RETURN (CRD$K_Repeat_Turing)
;    0253  1  END;      ! Routine HS$Advance_Diagnostic



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

69  44  5F  65  63  6E  61  76  64  41  20  2A  2A  2A  36  00009 P.AAB:  .ASCII  \6*** Advance_Diagnostic: Status bits for\  ;
75  74  61  74  53  20  3A  63  69  74  73  6F  6E  67  61  00018             ;
      72  6F  66  20  73  74  69  62  20  73  00027                 ;
2F  21  4C  58  21  20  3A  42  44  44  20  4C  58  21  20  00031    .ASCII  \ !XL DDB: !XL!/\          ;

            .EXTRN  CRD$GT_PASS, CRD$PRINT
            .EXTRN  CRD$GL_HS_CURRENT_STATE
            .EXTRN  MEN$FNCTST_TESTSTRT_TEXT
            .EXTRN  CRD$GB_CRD_DEBUG

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0004 00000     .ENTRY  HS$ADVANCE_DIAGNOSTIC, Save R2        ; 0156
            50 00000000G  EF  D0 00002     MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0          ; 0190
        00000000G  EF  D5 00009     TSTL    CRD$GL_HS_CURRENT_HSB_NUMB          ; 0183
            0A  12 0000F     BNEQ    1$              ;
        00000000G  EF 10    A0  D0 00011     MOVL    16(R0), CRD$GA_HS_CURRENT_HSB_PTR        ; 0190
            39  11 00019     BRB     3$              ; 0183
        00000000G  EF       20  C0 0001B 1$:   ADDL2   #32, CRD$GA_HS_CURRENT_HSB_PTR        ; 0199
            11    14    A0        06  E0 00022     BBS     #6, 20(R0), 2$          ; 0206
        00000000G  EF  9F 00027     PUSHAB  CRD$GT_PASS             ; 0207
            01  DD 0002D     PUSHL   #1              ;
            1A  DD 0002F     PUSHL   #26             ;
        00000000G  FF       03  FB 00031     CALLS   #3, @CRD$PRINT          ;
            50 00000000G  EF  D0 00038 2$:   MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0          ; 0209
    51      14    A0         01       04  EF 0003F     EXTZV   #4, #1, 20(R0), R1          ; 0211
    52      14    A0         01       00  EF 00045     EXTZV   #0, #1, 20(R0), R2          ;
            51       52  88 0004B     BISB2   R2, R1          ;
      14    A0          01        04       51  F0 0004E     INSV    R1, #4, #1, 20(R0)          ;
            50 00000000G  EF  D0 00054 3$:   MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0          ; 0214
```

```
      18   A0 00000000G  EF  D1 0005B    CMPL    CRD$GL_HS_CURRENT_HSB_NUMB, 24(R0)       ;
           09  12 00063    BNEQ    4$             ;
            00000000G  EF      10  D0 00065    MOVL    #16, CRD$GL_HS_CURRENT_STATE        ; 0220
           15  11 0006C    BRB     5$             ; 0214
            00000000G  EF  D6 0006E 4$:   INCL    CRD$GL_HS_CURRENT_HSB_NUMB        ; 0229
            00000000G  EF  DD 00074    PUSHL   CRD$GA_HS_CURRENT_HSB_PTR         ; 0235
           50  DD 0007A    PUSHL   R0             ;
            00000000G  EF      02  FB 0007C    CALLS   #2, MEN$FNCTST_TESTSTRT_TEXT        ;
           50 00000000G  EF  D0 00083 5$:   MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0        ; 0243
      14   A0 41   8F  8A 0008A    BICB2   #65, 20(R0)          ; 0244
           16 00000000G  EF  E9 0008F    BLBC    CRD$GB_CRD_DEBUG, 6$          ; 0246
      14   A0  DD 00096    PUSHL   20(R0)           ; 0249
           50  DD 00099    PUSHL   R0             ;
            00000000'  EF  9F 0009B    PUSHAB  P.AAB           ;
           01  DD 000A1    PUSHL   #1             ;
           1A  DD 000A3    PUSHL   #26            ;
            00000000G  FF      05  FB 000A5    CALLS   #5, @CRD$PRINT          ;
           50      02  D0 000AC 6$:   MOVL    #2, R0          ; 0252
            04 000AF    RET                 ; 0253

; Routine Size:  176 bytes,    Routine Base:  CODE + 001F
```

```
;    0254  1 %SBTTL 'HS$Load_General_Com Routine'
;    0255  1
;    0256  1 GLOBAL ROUTINE HS$Load_General_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0257  1
;    0258  1 !++
;    0259  1 ! Functional Description:
;    0260  1 !
;    0261  1 ! Describe function of routine
;    0262  1 !
;    0263  1 ! Formal Input Parameters:
;    0264  1 !
;    0265  1 ! None
;    0266  1 !
;    0267  1 ! Formal Output Parameters:
;    0268  1 !
;    0269  1 ! None
;    0270  1 !
;    0271  1 ! Side Effects:
;    0272  1 !
;    0273  1 ! None
;    0274  1 !
;    0275  1 ! Completion Codes:
;    0276  1 !
;    0277  1 ! None
;    0278  1 !--
```

```
;   0279  2 BEGIN     ! Routine HS$Load_General_Com
;   0280  2  RETURN (CRD$Load_General_Com (.CLI_Buffer_Length, .CLI_Buffer_Address));
;   0281  1 END;      ! Routine HS$Load_General_Com



          .EXTRN  CRD$LOAD_GENERAL_COM

            0000 00000     .ENTRY  HS$LOAD_GENERAL_COM, Save nothing       ; 0256
          7E 04    AC  7D 00002     MOVQ    CLI_BUFFER_LENGTH, -(SP)       ; 0280
      00000000G  EF       02  FB 00006     CALLS   #2, CRD$LOAD_GENERAL_COM       ;
       04 0000D     RET                 ; 0281

; Routine Size:  14 bytes,    Routine Base:  CODE + 00CF
```

```
;    0282  1 %SBTTL 'HS$Advance_General_Com Routine'
;    0283  1
;    0284  1 GLOBAL ROUTINE HS$Advance_General_Com =
;    0285  1
;    0286  1 !++
;    0287  1 ! Functional Description:
;    0288  1 !
;    0289  1 ! Describe function of routine
;    0290  1 !
;    0291  1 ! Formal Input Parameters:
;    0292  1 !
;    0293  1 ! None
;    0294  1 !
;    0295  1 ! Formal Output Parameters:
;    0296  1 !
;    0297  1 ! None
;    0298  1 !
;    0299  1 ! Side Effects:
;    0300  1 !
;    0301  1 ! None
;    0302  1 !
;    0303  1 ! Completion Codes:
;    0304  1 !
;    0305  1 ! None
;    0306  1 !--
```

```
;    0307  2 BEGIN     ! Routine HS$Advance_General_Com
;    0308  2  IF (.CRD$GL_Current_General_Com EQL (CRD$K_Numb_General_Commands - 1)) THEN
;    0309  2   CRD$GL_HS_Current_State = HS$K_State_Done_General_Coms
;    0310  2  ELSE
;    0311  2   CRD$GL_Current_General_Com = .CRD$GL_Current_General_Com + 1;
;    0312  2
;    0313  2  RETURN (CRD$K_Repeat_Turing);
;    0314  1 END;     ! Routine HS$Advance_General_Com



            .EXTRN  CRD$GL_CURRENT_GENERAL_COM

              0000 00000     .ENTRY  HS$ADVANCE_GENERAL_COM, Save nothing       ; 0284
            02 00000000G  EF  D1 00002     CMPL    CRD$GL_CURRENT_GENERAL_COM, #2       ; 0308
            09  12 00009     BNEQ    1$            ;
         00000000G  EF       07  D0 0000B     MOVL    #7, CRD$GL_HS_CURRENT_STATE        ; 0309
            06  11 00012     BRB     2$            ;
         00000000G  EF  D6 00014 1$:    INCL    CRD$GL_CURRENT_GENERAL_COM        ; 0311
            50       02  D0 0001A 2$:    MOVL    #2, R0           ; 0313
          04 0001D     RET                ; 0314

; Routine Size:  30 bytes,    Routine Base:  CODE + 00DD
```

```
;    0315  1 %SBTTL 'HS$Done_General_Coms Routine'
;    0316  1
;    0317  1 GLOBAL ROUTINE HS$Done_General_Coms =
;    0318  1
;    0319  1 !++
;    0320  1 ! Functional Description:
;    0321  1 !
;    0322  1 ! Describe function of routine
;    0323  1 !
;    0324  1 ! Formal Input Parameters:
;    0325  1 !
;    0326  1 ! None
;    0327  1 !
;    0328  1 ! Formal Output Parameters:
;    0329  1 !
;    0330  1 ! None
;    0331  1 !
;    0332  1 ! Side Effects:
;    0333  1 !
;    0334  1 ! None
;    0335  1 !
;    0336  1 ! Completion Codes:
;    0337  1 !
;    0338  1 ! None
;    0339  1 ! -
```

```
;   0340  2 BEGIN     ! Routine HS$Done_General_Coms
;   0341  2  RETURN (CRD$Done_General_Coms ());
;   0342  1 END;      ! Routine HS$Done_General_Coms



            .EXTRN  CRD$DONE_GENERAL_COMS

                0000 00000      .ENTRY  HS$DONE_GENERAL_COMS, Save nothing      ; 0317
        00000000G  EF       00  FB 00002      CALLS   #0, CRD$DONE_GENERAL_COMS       ; 0341
                04 00009      RET                       ; 0342

; Routine Size:  10 bytes,    Routine Base:  CODE + 00FB
```

```
;    0343  1 %SBTTL 'HS$Load_Load_Com Routine'
;    0344  1
;    0345  1 GLOBAL ROUTINE HS$Load_Load_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0346  1
;    0347  1 !++
;    0348  1 ! Functional Description:
;    0349  1 !
;    0350  1 ! Describe function of routine
;    0351  1 !
;    0352  1 ! Formal Input Parameters:
;    0353  1 !
;    0354  1 ! None
;    0355  1 !
;    0356  1 ! Formal Output Parameters:
;    0357  1 !
;    0358  1 ! None
;    0359  1 !
;    0360  1 ! Side Effects:
;    0361  1 !
;    0362  1 ! None
;    0363  1 !
;    0364  1 ! Completion Codes:
;    0365  1 !
;    0366  1 ! None
;    0367  1 !--
```

```
;    0368  2 BEGIN     ! Routine HS$Load_Load_Com
;    0369  2  MAP
;    0370  2   CRD$GT_DS_LOAD_Com : VECTOR [, BYTE];
;    0371  2
;    0372  2  MAP
;    0373  2   CRD$GA_HS_Current_HSB_Ptr : REF Hardware_Support_Vector;
;    0374  2
;    0375  2  BIND
;    0376  2   Diag_Name = (.CRD$GA_HS_Current_HSB_Ptr [CRD$K_Diagnostic_Name]) : VECTOR [, BYTE];
;    0377  2
;    0378  2  CH$COPY (
;    0379  2   .CRD$GT_DS_LOAD_Com [0], CH$PTR (CRD$GT_DS_LOAD_Com [1]),
;    0380  2   .Diag_Name [0],   CH$PTR (Diag_Name [1]),
;    0381  2   %C' ',
;    0382  2   .CLI_Buffer_Length,  CH$PTR (.CLI_Buffer_Address)
;    0383  2   );
;    0384  2
;    0385  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0386  2
;    0387  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0388  1 END;     ! Routine HS$Load_Load_Com


          .EXTRN  CRD$GT_DS_LOAD_COM

            07FC 00000     .ENTRY  HS$LOAD_LOAD_COM, Save R2,R3,R4,R5,R6,R7,-  ; 0345
           R8,R9,R10           ;
          58 00000000G   FF   D0 00002     MOVL    @CRD$GA_HS_CURRENT_HSB_PTR, R8        ; 0376
          5A 00000000G   EF   9A 00009     MOVZBL  CRD$GT_DS_LOAD_COM, R10         ; 0379
          59       68   9A 00010     MOVZBL  (R8), R9           ; 0380
          57 04    AC   D0 00013     MOVL    CLI_BUFFER_LENGTH, R7          ; 0382
          56 08    AC   D0 00017     MOVL    CLI_BUFFER_ADDRESS, R6         ;
    57           20 00000000G   EF       5A   2C 0001B     MOVC5   R10, CRD$GT_DS_LOAD_COM+1, #32, R7, (R6)     ;
          66       00024           ;
          0D   18 00025     BGEQ    1$             ;
          56       5A   C0 00027     ADDL2   R10, R6           ;
          57       5A   C2 0002A     SUBL2   R10, R7           ;
    57           20    01    A8       59   2C 0002D     MOVC5   R9, 1(R8), #32, R7, (R6)         ;
          66       00033           ;
          7E 04    AC   7D 00034 1$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)         ; 0385
       00000000V   EF       02   FB 00038     CALLS   #2, CRD$PRINT_DS_COMMAND         ;
          50 00500001    8F   D0 0003F     MOVL    #5242881, R0           ; 0387
           04 00046     RET                ; 0388

; Routine Size:  71 bytes,    Routine Base:  CODE + 0105
```

```
;    0389  1 %SBTTL 'HS$Load_Set_Flags_Com Routine'
;    0390  1
;    0391  1 GLOBAL ROUTINE HS$Load_Set_Flags_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0392  1
;    0393  1 !++
;    0394  1 ! Functional Description:
;    0395  1 !
;    0396  1 ! Describe function of routine
;    0397  1 !
;    0398  1 ! Formal Input Parameters:
;    0399  1 !
;    0400  1 ! None
;    0401  1 !
;    0402  1 ! Formal Output Parameters:
;    0403  1 !
;    0404  1 ! None
;    0405  1 !
;    0406  1 ! Side Effects:
;    0407  1 !
;    0408  1 ! None
;    0409  1 !
;    0410  1 ! Completion Codes:
;    0411  1 !
;    0412  1 ! None
;    0413  1 !--
```

```
;    0414  2 BEGIN     ! Routine HS$Load_Set_Flags_Com
;    0415  2  MAP
;    0416  2   CRD$GT_DS_SET_FLAGS_Com : VECTOR [, BYTE];
;    0417  2
;    0418  2  MAP
;    0419  2   CRD$GA_HS_Current_HSB_Ptr : REF Hardware_Support_Vector;
;    0420  2
;    0421  2  BIND
;    0422  2   Set_Flags_Args = (.CRD$GA_HS_Current_HSB_Ptr [CRD$K_Set_Flags_Args]) : VECTOR [, BYTE];
;    0423  2
;    0424  2  IF (.Set_Flags_Args [0] EQL 0) THEN
;    0425  3   RETURN (CRD$K_Repeat_Turing)
;    0426  2  ELSE
;    0427  2   CH$COPY (
;    0428  2     .CRD$GT_DS_SET_FLAGS_Com [0], CH$PTR (CRD$GT_DS_SET_FLAGS_Com [1]),
;    0429  2     .Set_Flags_Args [0],       CH$PTR (Set_Flags_Args [1]),
;    0430  2     %C' ',
;    0431  2     .CLI_Buffer_Length,  CH$PTR (.CLI_Buffer_Address)
;    0432  2     );
;    0433  2
;    0434  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0435  2
;    0436  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0437  1 END;     ! Routine HS$Load_Set_Flags_Com


        .EXTRN  CRD$GT_DS_SET_FLAGS_COM

          07FC 00000     .ENTRY  HS$LOAD_SET_FLAGS_COM, Save R2,R3,R4,R5,R6,-; 0391
         R7,R8,R9,R10            ;
        50 00000000G  EF  D0 00002     MOVL    CRD$GA_HS_CURRENT_HSB_PTR, R0        ; 0422
        57 04    A0  D0 00009     MOVL    4(R0), R7            ;
        67  95 0000D     TSTB    (R7)                 ; 0424
        04  12 0000F     BNEQ    1$                 ;
        50       02  D0 00011     MOVL    #2, R0             ; 0425
     04 00014     RET                 ;
        5A 00000000G  EF  9A 00015 1$:   MOVZBL  CRD$GT_DS_SET_FLAGS_COM, R10        ; 0428
        59       67  9A 0001C     MOVZBL  (R7), R9           ; 0429
        58 04    AC  D0 0001F     MOVL    CLI_BUFFER_LENGTH, R8          ; 0431
        56 08    AC  D0 00023     MOVL    CLI_BUFFER_ADDRESS, R6         ;
    58           20 00000000G  EF       5A  2C 00027     MOVC5   R10, CRD$GT_DS_SET_FLAGS_COM+1, #32, R8, -   ;
        66       00030     (R6)               ;
        0D  18 00031     BGEQ    2$                 ;
        56       5A  C0 00033     ADDL2   R10, R6            ;
        58       5A  C2 00036     SUBL2   R10, R8            ;
    58           20    01    A7       59  2C 00039     MOVC5   R9, 1(R7), #32, R8, (R6)         ;
        66       0003F                ;
        7E 04    AC  7D 00040 2$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)         ; 0434
     00000000V  EF       02  FB 00044     CALLS   #2, CRD$PRINT_DS_COMMAND         ;
        50 00500001    8F  D0 0004B     MOVL    #5242881, R0           ; 0436
     04 00052     RET                 ; 0437

; Routine Size:  83 bytes,    Routine Base:  CODE + 014C
```

```
;    0438  1 %SBTTL 'HS$Load_Set_Event_Com Routine'
;    0439  1
;    0440  1 GLOBAL ROUTINE HS$Load_Set_Event_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0441  1
;    0442  1 !++
;    0443  1 ! Functional Description:
;    0444  1 !
;    0445  1 ! Describe function of routine
;    0446  1 !
;    0447  1 ! Formal Input Parameters:
;    0448  1 !
;    0449  1 ! None
;    0450  1 !
;    0451  1 ! Formal Output Parameters:
;    0452  1 !
;    0453  1 ! None
;    0454  1 !
;    0455  1 ! Side Effects:
;    0456  1 !
;    0457  1 ! None
;    0458  1 !
;    0459  1 ! Completion Codes:
;    0460  1 !
;    0461  1 ! None
;    0462  1 !--
```

```
;    0463  2 BEGIN     ! Routine HS$Load_Set_Event_Com
;    0464  2  MAP
;    0465  2   CRD$GT_DS_SET_EVENT_Com : VECTOR [, BYTE];
;    0466  2
;    0467  2  MAP
;    0468  2   CRD$GA_HS_Current_HSB_Ptr : REF Hardware_Support_Vector;
;    0469  2
;    0470  2  BIND
;    0471  2   Set_Event_Args = (.CRD$GA_HS_Current_HSB_Ptr [CRD$K_Set_Event_Args]) : VECTOR [, BYTE];
;    0472  2
;    0473  2  IF (.Set_Event_Args [0] EQL 0) THEN
;    0474  3   RETURN (CRD$K_Repeat_Turing)
;    0475  2  ELSE
;    0476  2   CH$COPY (
;    0477  2    .CRD$GT_DS_SET_EVENT_Com [0], CH$PTR (CRD$GT_DS_SET_EVENT_Com [1]),
;    0478  2    .Set_Event_Args [0],       CH$PTR (Set_Event_Args [1]),
;    0479  2    %C' ',
;    0480  2    .CLI_Buffer_Length,  CH$PTR (.CLI_Buffer_Address)
;    0481  2    );
;    0482  2
;    0483  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0484  2
;    0485  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0486  1 END;      ! Routine HS$Load_Set_Event_Com


          .EXTRN  CRD$GT_DS_SET_EVENT_COM

            07FC 00000     .ENTRY  HS$LOAD_SET_EVENT_COM, Save R2,R3,R4,R5,R6,-; 0440
           R7,R8,R9,R10            ;
          50 00000000G  EF  D0 00002     MOVL    CRD$GA_HS_CURRENT_HSB_PTR, R0          ; 0471
          57 08    A0  D0 00009     MOVL    8(R0), R7              ;
          67  95 0000D     TSTB    (R7)                   ; 0473
          04  12 0000F     BNEQ    1$                ;
          50       02  D0 00011     MOVL    #2, R0                 ; 0474
       04 00014     RET                     ;
          5A 00000000G  EF  9A 00015 1$:   MOVZBL  CRD$GT_DS_SET_EVENT_COM, R10          ; 0477
          59       67  9A 0001C     MOVZBL  (R7), R9               ; 0478
          58 04    AC  D0 0001F     MOVL    CLI_BUFFER_LENGTH, R8          ; 0480
          56 08    AC  D0 00023     MOVL    CLI_BUFFER_ADDRESS, R6         ;
    58          20 00000000G  EF       5A  2C 00027     MOVC5   R10, CRD$GT_DS_SET_EVENT_COM+1, #32, R8, -   ;
          66      00030     (R6)                   ;
          0D  18 00031     BGEQ    2$                ;
          56       5A  C0 00033     ADDL2   R10, R6                ;
          58       5A  C2 00036     SUBL2   R10, R8                ;
    58          20    01    A7       59  2C 00039     MOVC5   R9, 1(R7), #32, R8, (R6)          ;
          66      0003F                  ;
          7E 04    AC  7D 00040 2$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)           ; 0483
       00000000V  EF       02  FB 00044     CALLS   #2, CRD$PRINT_DS_COMMAND           ;
          50 00500001    8F  D0 0004B     MOVL    #5242881, R0           ; 0485
       04 00052     RET                     ; 0486

; Routine Size:  83 bytes,    Routine Base:  CODE + 019F
```

```
;    0487  1 %SBTTL 'HS$Load_Select_Com Routine'
;    0488  1
;    0489  1 GLOBAL ROUTINE HS$Load_Select_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0490  1
;    0491  1 !++
;    0492  1 ! Functional Description:
;    0493  1 !
;    0494  1 ! Describe function of routine
;    0495  1 !
;    0496  1 ! Formal Input Parameters:
;    0497  1 !
;    0498  1 ! None
;    0499  1 !
;    0500  1 ! Formal Output Parameters:
;    0501  1 !
;    0502  1 ! None
;    0503  1 !
;    0504  1 ! Side Effects:
;    0505  1 !
;    0506  1 ! None
;    0507  1 !
;    0508  1 ! Completion Codes:
;    0509  1 !
;    0510  1 ! None
;    0511  1 !--
```

```
;    0512  2 BEGIN     ! Routine HS$Load_Select_Com
;    0513  2  MAP
;    0514  2   CRD$GA_HS_Current_DDB_Ptr : REF Device_Data_Block_Structure;
;    0515  2
;    0516  2  MAP
;    0517  2   CRD$GT_DS_SELECT_Com : VECTOR [, BYTE];
;    0518  2
;    0519  2  LOCAL
;    0520  2   Device_Name_ASCIC_Ptr;
;    0521  2
;    0522  2  Device_Name_ASCIC_Ptr = CRD$Get_Device_Name (.CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_PTable_Entry]);
;    0523  2
;    0524  3   BEGIN
;    0525  3   MAP
;    0526  3    Device_Name_ASCIC_Ptr : REF VECTOR [, BYTE];
;    0527  3
;    0528  3   CH$COPY (
;    0529  3    .CRD$GT_DS_SELECT_Com [0],  CH$PTR (CRD$GT_DS_SELECT_Com [1]),
;    0530  3    .Device_Name_ASCIC_Ptr [0], CH$PTR (Device_Name_ASCIC_Ptr [1]),
;    0531  3    %C' ',
;    0532  3    .CLI_Buffer_Length,    CH$PTR (.CLI_Buffer_Address)
;    0533  3    );
;    0534  2   END;
;    0535  2
;    0536  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0537  2
;    0538  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0539  1 END;     ! Routine HS$Load_Select_Com


        .EXTRN  CRD$GT_DS_SELECT_COM
        .EXTRN  CRD$GET_DEVICE_NAME

          07FC 00000     .ENTRY  HS$LOAD_SELECT_COM, Save R2,R3,R4,R5,R6,R7,-; 0489
         R8,R9,R10              ;
        50 00000000G  EF  D0 00002     MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0          ; 0522
    08  A0  DD 00009     PUSHL   8(R0)                ;
     00000000G  EF       01  FB 0000C     CALLS   #1, CRD$GET_DEVICE_NAME          ;
        58       50  D0 00013     MOVL    R0, DEVICE_NAME_ASCIC_PTR          ;
        5A 00000000G  EF  9A 00016     MOVZBL  CRD$GT_DS_SELECT_COM, R10          ; 0529
        59       68  9A 0001D     MOVZBL  (DEVICE_NAME_ASCIC_PTR), R9          ; 0530
        57 04    AC  D0 00020     MOVL    CLI_BUFFER_LENGTH, R7          ; 0532
        56 08    AC  D0 00024     MOVL    CLI_BUFFER_ADDRESS, R6          ;
    57           20 00000000G  EF       5A  2C 00028     MOVC5   R10, CRD$GT_DS_SELECT_COM+1, #32, R7, (R6)  :
        66       00031              ;
        0D  18 00032     BGEQ    1$             ;
        56       5A  C0 00034     ADDL2   R10, R6          ;
        57       5A  C2 00037     SUBL2   R10, R7          ;
    57           20   01   A8       59  2C 0003A     MOVC5   R9, 1(DEVICE_NAME_ASCIC_PTR), #32, R7, (R6) :
        66       00040              ;
        7E 04    AC  7D 00041 1$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)          ; 0536
     00000000V  EF       02  FB 00045     CALLS   #2, CRD$PRINT_DS_COMMAND          ;
        50 00500001    8F  D0 0004C     MOVL    #5242881, R0          ; 0538
      04 00053     RET                ; 0539
```

```
; Routine Size:  84 bytes,    Routine Base:  CODE + 01F2
```

```
;    0540  1 %SBTTL 'HS$Load_Start_Com Routine'
;    0541  1
;    0542  1 GLOBAL ROUTINE HS$Load_Start_Com (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0543  1
;    0544  1 !++
;    0545  1 ! Functional Description:
;    0546  1 !
;    0547  1 ! Describe function of routine
;    0548  1 !
;    0549  1 ! Formal Input Parameters:
;    0550  1 !
;    0551  1 ! None
;    0552  1 !
;    0553  1 ! Formal Output Parameters:
;    0554  1 !
;    0555  1 ! None
;    0556  1 !
;    0557  1 ! Side Effects:
;    0558  1 !
;    0559  1 ! None
;    0560  1 !
;    0561  1 ! Completion Codes:
;    0562  1 !
;    0563  1 ! None
;    0564  1 !--
```

```
;    0565  2 BEGIN     ! Routine HS$Load_Start_Com
;    0566  2  MAP
;    0567  2   CRD$GT_DS_START_Com : VECTOR [, BYTE];
;    0568  2
;    0569  2  MAP
;    0570  2   CRD$GA_HS_Current_HSB_Ptr : REF Hardware_Support_Vector;
;    0571  2
;    0572  2  BIND
;    0573  2   Start_Qualifiers = (.CRD$GA_HS_Current_HSB_Ptr [CRD$K_Start_Qualifiers]) : VECTOR [, BYTE];
;    0574  2
;    0575  2  CH$COPY (
;    0576  2   .CRD$GT_DS_START_Com [0], CH$PTR (CRD$GT_DS_START_Com [1]),
;    0577  2   .Start_Qualifiers [0],   CH$PTR (Start_Qualifiers [1]),
;    0578  2   %C' ',
;    0579  2   .CLI_Buffer_Length,   CH$PTR (.CLI_Buffer_Address)
;    0580  2   );
;    0581  2
;    0582  2  CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0583  2
;    0584  2  $CRD_Return_Length_Status (CRD$K_Success);
;    0585  1 END;     ! Routine HS$Load_Start_Com



          .EXTRN  CRD$GT_DS_START_COM

            07FC 00000     .ENTRY  HS$LOAD_START_COM, Save R2,R3,R4,R5,R6,R7,- ; 0542
           R8,R9,R10               ;
          50 00000000G  EF  D0 00002     MOVL    CRD$GA_HS_CURRENT_HSB_PTR, R0          ; 0573
          58 10    A0  D0 00009     MOVL    16(R0), R8             ;
          5A 00000000G  EF  9A 0000D     MOVZBL  CRD$GT_DS_START_COM, R10          ; 0576
          59       68  9A 00014     MOVZBL  (R8), R9               ; 0577
          57 04    AC  D0 00017     MOVL    CLI_BUFFER_LENGTH, R7          ; 0579
          56 08    AC  D0 0001B     MOVL    CLI_BUFFER_ADDRESS, R6         ;
    57          20 00000000G  EF       5A  2C 0001F     MOVC5   R10, CRD$GT_DS_START_COM+1, #32, R7, (R6)    ;
          66       00028                ;
          0D  18 00029     BGEQ    1$             ;
          56       5A  C0 0002B     ADDL2   R10, R6            ;
          57       5A  C2 0002E     SUBL2   R10, R7            ;
    57          20   01    A8       59  2C 00031     MOVC5   R9, 1(R8), #32, R7, (R6)         ;
          66       00037                ;
          7E 04    AC  7D 00038 1$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)         ; 0582
       00000000V  EF       02  FB 0003C     CALLS   #2, CRD$PRINT_DS_COMMAND         ;
          50 00500001   8F  D0 00043     MOVL    #5242881, R0           ; 0584
        04 0004A     RET                 ; 0585

; Routine Size:  75 bytes,    Routine Base:  CODE + 0246
```

```
;    0586  1 %SBTTL 'HS$Change_Table Routine'
;    0587  1
;    0588  1 GLOBAL ROUTINE HS$Change_Table =
;    0589  1
;    0590  1 !++
;    0591  1 ! Functional Description:
;    0592  1 !
;    0593  1 ! Describe function of routine
;    0594  1 !
;    0595  1 ! Formal Input Parameters:
;    0596  1 !
;    0597  1 ! None
;    0598  1 !
;    0599  1 ! Formal Output Parameters:
;    0600  1 !
;    0601  1 ! None
;    0602  1 !
;    0603  1 ! Side Effects:
;    0604  1 !
;    0605  1 ! None
;    0606  1 !
;    0607  1 ! Completion Codes:
;    0608  1 !
;    0609  1 ! None
;    0610  1 !--
```

```
;    0611  2 BEGIN     ! Routine HS$Change_Table
;    0612  2  MAP
;    0613  2   CRD$GA_HS_Current_DDB_Ptr : REF Device_Data_Block_Structure;
;    0614  2
;    0615  2  !+
;    0616  2  ! Go back to the Test Queue State Table. Do any cleanups that are necessary here.
;    0617  2  !-
;    0618  2
;    0619  2  CRD$GL_HS_Current_State = HS$K_State_Init_HS;
;    0620  2      ! Reset this to its start state
;    0621  2
;    0622  2  CRD$GB_Current_State_Table = MEN$K_TQ_State_Table;
;    0623  2      ! Use the TQ State Table now
;    0624  2
;    0625  3  RETURN (CRD$K_Repeat_Turing)
;    0626  1 END;     ! Routine HS$Change_Table



          .EXTRN  CRD$GB_CURRENT_STATE_TABLE

                0000 00000     .ENTRY  HS$CHANGE_TABLE, Save nothing          ; 0588
       00000000G  EF       03  D0 00002     MOVL     #3, CRD$GL_HS_CURRENT_STATE          ; 0619
       00000000G  EF       01  90 00009     MOVB     #1, CRD$GB_CURRENT_STATE_TABLE         ; 0622
           50       02  D0 00010     MOVL     #2, R0              ; 0625
        04 00013     RET                 ; 0626

; Routine Size:  20 bytes,    Routine Base:  CODE + 0291
```

```
;    0627  1 %SBTTL 'HS$Load_Error Routine'
;    0628  1
;    0629  1 GLOBAL ROUTINE HS$Load_Error =
;    0630  1
;    0631  1 !++
;    0632  1 ! Functional Description:
;    0633  1 !
;    0634  1 ! Describe function of routine
;    0635  1 !
;    0636  1 ! Formal Input Parameters:
;    0637  1 !
;    0638  1 ! None
;    0639  1 !
;    0640  1 ! Formal Output Parameters:
;    0641  1 !
;    0642  1 ! None
;    0643  1 !
;    0644  1 ! Side Effects:
;    0645  1 !
;    0646  1 ! None
;    0647  1 !
;    0648  1 ! Completion Codes:
;    0649  1 !
;    0650  1 ! None
;    0651  1 !--
```

```
;   0652  2 BEGIN     ! Routine HS$Load_Error
;   0653  2  MAP
;   0654  2   CRD$GA_HS_Current_DDB_Ptr : REF Device_Data_Block_Structure;
;   0655  2
;   0656  2  CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed> = True;
;   0657  2
;   0658  2  CRD$GA_Current_Diagnostic = .CRD$GA_HS_Current_DDB_Ptr;
;   0659  2  RETURN (CRD$Load_Error ()); ! Let it do the work
;   0660  1 END;     ! Routine HS$Load_Error



          .EXTRN  CRD$GA_CURRENT_DIAGNOSTIC
          .EXTRN  CRD$LOAD_ERROR

            0000 C0000     .ENTRY  HS$LOAD_ERROR, Save nothing          ; 0629
           50 00000000G  EF  D0 00002     MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0          ; 0656
      14   A0 40   8F  88 00009     BISB2   #64, 20(R0)             ;
       00000000G  EF      50  D0 0000E     MOVL    R0, CRD$GA_CURRENT_DIAGNOSTIC          ; 0658
       00000000G  EF      00  FB 00015     CALLS   #0, CRD$LOAD_ERROR          ; 0659
         04 0001C     RET                 ; 0660

; Routine Size:  29 bytes,    Routine Base:  CODE + 02A5
```

```
;    0661  1 %SBTTL 'HS$Start_Error Routine'
;    0662  1
;    0663  1 GLOBAL ROUTINE HS$Start_Error =
;    0664  1
;    0665  1 !++
;    0666  1 ! Functional Description:
;    0667  1 !
;    0668  1 ! Describe function of routine
;    0669  1 !
;    0670  1 ! Formal Input Parameters:
;    0671  1 !
;    0672  1 ! None
;    0673  1 !
;    0674  1 ! Formal Output Parameters:
;    0675  1 !
;    0676  1 ! None
;    0677  1 !
;    0678  1 ! Side Effects:
;    0679  1 !
;    0680  1 ! None
;    0681  1 !
;    0682  1 ! Completion Codes:
;    0683  1 !
;    0684  1 ! None
;    0685  1 !--
```

```
;   0686  2 BEGIN     ! Routine HS$Start_Error
;   0687  2  MAP
;   0688  2   CRD$GA_HS_Current_DDB_Ptr : REF Device_Data_Block_Structure;
;   0689  2
;   0690  2  CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed> = True;
;   0691  2
;   0692  2  CRD$GA_Current_Diagnostic = .CRD$GA_HS_Current_DDB_Ptr;
;   0693  2  RETURN (CRD$Start_Error ()); ! Let it do the work
;   0694  1 END;     ! Routine HS$Start_Error



        .EXTRN  CRD$START_ERROR

          0000 00000     .ENTRY  HS$START_ERROR, Save nothing        ; 0663
         50 00000000G  EF  D0 00002     MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0        ; 0690
    14   A0 40    8F  88 00009     BISB2   #64, 20(R0)          ;
     00000000G  EF       50  D0 0000E     MOVL    R0, CRD$GA_CURRENT_DIAGNOSTIC        ; 0692
     00000000G  EF       00  FB 00015     CALLS   #0, CRD$START_ERROR          ; 0693
      04 0001C     RET                 ; 0694

; Routine Size:  29 bytes,    Routine Base:  CODE + 02C2
```

```
;    0695  1 %SBTTL 'HS$Diagnostic_Error Routine'
;    0696  1
;    0697  1 GLOBAL ROUTINE HS$Diagnostic_Error (TypeCode, Error_Message_Length, Error_Message_Address) -
;    0698  1
;    0699  1 !++
;    0700  1 ! Functional Description:
;    0701  1 !
;    0702  1 ! Describe function of routine
;    0703  1 !
;    0704  1 ! Formal Input Parameters:
;    0705  1 !
;    0706  1 ! None
;    0707  1 !
;    0708  1 ! Formal Output Parameters:
;    0709  1 !
;    0710  1 ! None
;    0711  1 !
;    0712  1 ! Side Effects:
;    0713  1 !
;    0714  1 ! None
;    0715  1 !
;    0716  1 ! Completion Codes:
;    0717  1 !
;    0718  1 ! None
;    0719  1 !--
```

```
;    0720  2 BEGIN     ! Routine HS$Diagnostic_Error
;    0721  2  MAP
;    0722  2   CRD$GA_HS_Current_DDB_Ptr : REF Device_Data_Block_Structure;
;    0723  2
;    0724  2  !+
;    0725  2  ! Indicate that this is a bad device, we did not finish testing, and that a message was printed.
;    0726  2  !-
;    0727  2
;    0728  2  CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Device_Bad>       = True;
;    0729  2  CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed>   = True;
;    0730  2  CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete> = False;
;    0731  2
;    0732  2  !+
;    0733  2  ! Print the  *FAILED*  message.
;    0734  2  !-
;    0735  2
;    0736  2  $CRD_Print (CRD$GT_Fail);
;    0737  2
;    0738  2  IF (.DSA$V_CRD_Trace) THEN
;    0739  3   BEGIN
;    0740  3   BIND
;    0741  3    Error_Text_Ptr = ((.Error_Message_Address) + 1),
;    0742  3    Error_Text_Len = (.Error_Message_Length - 1);
;    0743  3
; P  0744  3   $CRD_Print ($ASCIC ('*** The error message text:!/!AD!/*** End of the error message text!/'),
;    0745  3    Error_Text_Len, Error_Text_Ptr);
;    0746  2   END;
;    0747  2
;    0748  2  !+
;    0749  2  ! Abort the currently running diagnostic.
;    0750  2  !-
;    0751  2
;    0752  2  $DS_Abort (Arg = Program);
;    0753  2
;    0754  2  !+
;    0755  2  ! NOTE: the above will not return to here. Do the following for BLISS's sake.
;    0756  2  !-
;    0757  2
;    0758  3  RETURN (CRD$K_Return_Success)
;    0759  1 END;     ! Routine HS$Diagnostic_Error


            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

20  72  6F  72  72  65  20  65  68  54  20  2A  2A  2A  45  00040 P.AAC:  .ASCII  \E*** The error message text:!/!AD./*** E\  ;
2F  21  3A  74  78  65  74  20  65  67  61  73  73  65  6D  0004F             ;
      45  20  2A  2A  2A  2F  21  44  41  21  0005E                 ;
72  6F  72  72  65  20  65  68  74  20  66  6F  20  64  6E  00068    .ASCII  \nd of the error message text!/\      ;
2F  21  74  78  65  74  20  65  67  61  73  73  65  6D  20  00077             ;

            .EXTRN  CRD$GT_FAIL, DS$ABORT

            .PSECT  CODE,NOWRT,  SHR,  PIC,2
```

```
                0000 00000     .ENTRY  HS$DIAGNOSTIC_ERROR, Save nothing       ; 0697
            50 00000000G  EF   D0 00002     MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0       ; 0728
      14    A0 50    8F   88 00009     BISB2   #80, 20(R0)                ; 0729
      14    A0       20   8A 0000E     BICB2   #32, 20(R0)                ; 0730
       00000000G  EF   9F 00012     PUSHAB  CRD$GT_FAIL                ; 0736
            01   DD 00018     PUSHL   #1                 ;
            1A   DD 0001A     PUSHL   #26                ;
       00000000G  FF       03   FB 0001C     CALLS   #3, @CRD$PRINT             ;
            1D 0000FE02   9F       01   E1 00023     BBC     #1, @#^X0000FE02, 1$       ; 0738
            51    0C    AC       01   C1 0002B     ADDL3   #1, ERROR_MESSAGE_ADDRESS, R1       ; 0741
            50    08    AC       01   C3 00030     SUBL3   #1, ERROR_MESSAGE_LENGTH, R0        ; 0742
            03   BB 00035     PUSHR   #^M<R0,R1>             ; 0745
       00000000'  EF   9F 00037     PUSHAB  P.AAC              ;
            01   DD 0003D     PUSHL   #1                 ;
            1A   DD 0003F     PUSHL   #26                ;
       00000000G  FF       05   FB 00041     CALLS   #5, @CRD$PRINT             ;
       00000000G  9F       00   FB 00048 1$:   CALLS   #0, @#DS$ABORT           ; 0752
            50       01   D0 0004F     MOVL    #1, R0             ; 0758
        04 00052     RET                ; 0759

; Routine Size:  83 bytes,    Routine Base:  CODE + 02DF
```

```
;    0760  1 %SBTTL 'HS$Preparation_Error Routine'
;    0761  1
;    0762  1 GLOBAL ROUTINE HS$Preparation_Error =
;    0763  1
;    0764  1 !++
;    0765  1 ! Functional Description:
;    0766  1 !
;    0767  1 ! Describe function of routine
;    0768  1 !
;    0769  1 ! Formal Input Parameters:
;    0770  1 !
;    0771  1 ! None
;    0772  1 !
;    0773  1 ! Formal Output Parameters:
;    0774  1 !
;    0775  1 ! None
;    0776  1 !
;    0777  1 ! Side Effects:
;    0778  1 !
;    0779  1 ! None
;    0780  1 !
;    0781  1 ! Completion Codes:
;    0782  1 !
;    0783  1 ! None
;    0784  1 !--
```

```
;    0785  2 BEGIN     ! Routine HS$Preparation_Error
;    0786  2  !+
;    0787  2  ! Note:              ![01]
;    0788  2  ! CRD$GA_User_Error_Text          ![01]
;    0789  2  !  -> DS$GA_User_Err_Text         ![01]
;    0790  2  !   = Address of user's MSGADR ASCIC error message string    ![01]
;    0791  2  !       or = 0 (if no MSGADR string)       ![01]
;    0792  2  !-
;    0793  2
;    0794  2  BIND
;    0795  2   Addr_Of_Addr_Of_Error_Text =        ![01]
;    0796  2    (.CRD$GA_User_Error_Text);  ! I.e., address of DS$GA_User_Err_Text   ![01]
;    0797  2
;    0798  2  MAP
;    0799  2   CRD$GA_HS_Current_DDB_Ptr : REF Device_Data_Block_Structure;
;    0800  2
;    0801  2  CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete> = False;
;    0802  2  CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Message_Printed>   = True;
;    0803  2  CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Preparation_Error> = True;
;    0804  2
;    0805  2  !+
;    0806  2  ! From the above comment, pass either 0 or the address of the user's error text string.   ![01]
;    0807  2  !-
;    0808  2
;    0809  2  MEN$Preparation_Err_Text (.Addr_Of_Addr_Of_Error_Text, .CRD$GA_HS_Current_DDB_Ptr);  ![01]
;    0810  2
;    0811  2  $DS_Abort (Arg = Program); ! Abort the currently running diagnostic
;    0812  2
;    0813  3  RETURN (CRD$K_Repeat_Turing)
;    0814  1 END;     ! Routine HS$Preparation_Error



          .EXTRN  CRD$GA_USER_ERROR_TEXT
          .EXTRN  MEN$PREPARATION_ERR_TEXT

            0000 00000     .ENTRY  HS$PREPARATION_ERROR, Save nothing       ; 0762
          50 00000000G  EF  D0 00002     MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0        ; 0801
     14   A0       20  8A 00009     BICB2   #32, 20(R0)          ;
     14   A0 42    8F  88 0000D     BISB2   #66, 20(R0)          ; 0803
          50  DD 00012     PUSHL   R0           ; 0809
      00000000G  FF  DD 00014     PUSHL   @CRD$GA_USER_ERROR_TEXT        ;
      00000000G  EF       02  FB 0001A     CALLS   #2, MEN$PREPARATION_ERR_TEXT        ;
      00000000G  9F       00  FB 00021     CALLS   #0, @#DS$ABORT          ; 0811
          50       02  D0 00028     MOVL    #2, R0          ; 0813
       04 0002B     RET                 ; 0814

; Routine Size:  44 bytes,    Routine Base:  CODE + 0332
```

```
;    0815  1 %SBTTL 'HS$Internal_Error Routine'
;    0816  1
;    0817  1 GLOBAL ROUTINE HS$Internal_Error (TypeCode, Length, Address) =
;    0818  1
;    0819  1 !++
;    0820  1 ! Functional Description:
;    0821  1 !
;    0822  1 ! Describe function of routine
;    0823  1 !
;    0824  1 ! Formal Input Parameters:
;    0825  1 !
;    0826  1 ! None
;    0827  1 !
;    0828  1 ! Formal Output Parameters:
;    0829  1 !
;    0830  1 ! None
;    0831  1 !
;    0832  1 ! Side Effects:
;    0833  1 !
;    0834  1 ! None
;    0835  1 !
;    0836  1 ! Completion Codes:
;    0837  1 !
;    0838  1 ! None
;    0839  1  --
```

```
;   0840  2 BEGIN     ! Routine HS$Internal_Error
;   0841  2  MEN$Menu_Test_Internal_Error (CRD$K_TQ_Internal_Error, .TypeCode, .Length, .Address);
;   0842  2  CRD$Stop (MENU$K_Exit_Internal_Error);
;   0843  3  RETURN (CRD$K_Return_Failure)
;   0844  1 END;     ! Routine HS$Internal_Error



          .EXTRN  MEN$MENU_TEST_INTERNAL_ERROR
          .EXTRN  CRD$STOP

            0000 00000     .ENTRY  HS$INTERNAL_ERROR, Save nothing       ; 0817
          7E 08    AC  7D 00002    MOVQ    LENGTH, -(SP)                ; 0841
       04 AC  DD 00006    PUSHL   TYPECODE                :
          7E       07  CE 00009    MNEGL   #7, -(SP)              ;
        00000000G  EF       04  FB 0000C    CALLS   #4, MEN$MENU_TEST_INTERNAL_ERROR       ;
          0D  DD 00013    PUSHL   #13                  ; 0842
        00000000G  EF       01  FB 00015    CALLS   #1, CRD$STOP           ;
          50  D4 0001C    CLRL    R0                 ; 0843
           04 0001E    RET                     ; 0844

; Routine Size:  31 bytes,    Routine Base:  CODE + 035E
```

```
;    0845  1 %SBTTL 'CRD$Print_DS_Command Routine'
;    0846  1
;    0847  1 GLOBAL ROUTINE CRD$Print_DS_Command (CLI_Buffer_Length, CLI_Buffer_Address) : NOVALUE =
;    0848  1
;    0849  1 !++
;    0850  1 ! Functional Description:
;    0851  1 !
;    0852  1 ! Print the DS command that was loaded prior to this routine being called, iff the
;    0853  1 ! DSA$V_CRD_Trace flag is set.
;    0854  1 !
;    0855  1 ! Formal Input Parameters:
;    0856  1 !
;    0857  1 ! CLI_Buffer_Length: The length of the CLI command buffer
;    0858  1 ! CLI_Buffer_Address: The address of the CLI command buffer
;    0859  1 !
;    0860  1 ! Formal Output Parameters:
;    0861  1 !
;    0862  1 ! None
;    0863  1 !
;    0864  1 ! Side Effects:
;    0865  1 !
;    0866  1 ! None
;    0867  1 !
;    0868  1 ! Completion Codes:
;    0869  1 !
;    0870  1 ! None
;    0871  1 !--
```

```
;   0872  2 BEGIN     ! Routine CRD$Print_DS_Command
;   0873  2  IF (.DSA$V_CRD_Trace) THEN
;   0874  3   BEGIN
;   0875  3   $CRD_Print (CRD$GT_DS_Command_Out, .CLI_Buffer_Length, .CLI_Buffer_Address);
;   0876  2   END;
;   0877  1 END;     ! Routine CRD$Print_DS_Command



          .EXTRN  CRD$GT_DS_COMMAND_OUT

            0000 00000     .ENTRY  CRD$PRINT_DS_COMMAND, Save nothing       ; 0847
         15 0000FE02   9F       01  E1 00002     BBC     #1, a#^X0000FE02, 1$          ; 0873
         7E 04    AC  7D 0000A     MOVQ    CLI_BUFFER_LENGTH, -(SP)         ; 0875
      00000000G  EF  9F 0000E     PUSHAB  CRD$GT_DS_COMMAND_OUT          ;
         01  DD 00014     PUSHL   #1              ;
         1A  DD 00016     PUSHL   #26             ;
      00000000G  FF       05  FB 00018     CALLS   #5, aCRD$PRINT          ;
       04 0001F 1$:   RET                     ; 0877

; Routine Size:  32 bytes,    Routine Base:  CODE + 037D
```

```
;   0878  1 END     ! End of CRDHSACT module
;   0879  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          134  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          925  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$ASCIC    Macro    Lib02   248     744
$CRD_LITERALS   Macro    Lib03    93
$CRD_PRINT    Macro    Lib03   207     248     736     744     875
$CRD_RETURN_LENGTH_STATUS Macro    Lib03   387     436     485     538     584
$DS_ABORT    KeyWMacr Lib02   752     811
$DS_DSADEF    Macro    Lib02    92
$DS_TYPEDEF    Macro    Lib02   207     249     736     745     875
$EQULST    Macro    Lib04    93     207     249     736     745     875
$ER    Comptime   109
$MODULE    Bind       109
ADDRESS    RoutForm   817   841.
ADDR_OF_ADDR_OF_ERROR_TEXT Bind       795   809.
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     93
AUTO$K_EXIT_CONTROL_C  Literal     93
AUTO$K_EXIT_DUMMY_2  Literal     93
AUTO$K_EXIT_DUMMY_3  Literal     93
AUTO$K_EXIT_ERRORS_COMPLETE Literal     93
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     93
AUTO$K_EXIT_INTERNAL_ERROR Literal     93
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     93
AUTO$K_EXIT_LAST_REASON  Literal     93    93
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     93
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     93
AUTO$K_EXIT_NOERRORS_PREP Literal     93
CLI_BUFFER_ADDRESS  RoutForm   256   280.
    RoutForm   345   382a=   385.
    RoutForm   391   431a=   434.
    RoutForm   440   480a=   483.
    RoutForm   489   532a=   536.
    RoutForm   542   579a=   582.
    RoutForm   847   875.
CLI_BUFFER_LENGTH  RoutForm   256   280.
    RoutForm   345   382.    385.
    RoutForm   391   431.    434.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    RoutForm   440   480.    483.
    RoutForm   489   532.    536.
    RoutForm   542   579.    582.
    RoutForm   847   875.
CRD$DONE_GENERAL_COMS  ExtRout  Lib03   341c
CRD$GA_CURRENT_DIAGNOSTIC External Lib03   658=    692=
CRD$GA_HS_CURRENT_DDB_PTR External Lib03   137m    150.
    Map         181   190a.   206a.   209a=   210a.   211a.   214a.   235.    243a=   244a=   249.
              249a.
    Map         514   522a.
    Map         613
    Map         654   656a=   658.
    Map         688   690a=   692.
    Map         722   728a=   729a=   730a=
    Map         799   801a=   802a=   803a=   809.
CRD$GA_HS_CURRENT_HSB_PTR External Lib03   143=    190=    199=    199.    235.
    Map         373   376a.
    Map         419   422a.
    Map         468   471a.
    Map         570   573a.
CRD$GA_USER_ERROR_TEXT  External Lib03   796.
CRD$GB_CRD_DEBUG  External Lib03   246.
CRD$GB_CURRENT_STATE_TABLE External Lib03   622=
CRD$GET_DEVICE_NAME  ExtRout  Lib03   522c
CRD$GL_CURRENT_GENERAL_COM External Lib03   308.    311=    311.
CRD$GL_HS_CURRENT_HSB_NUMB External Lib03   144=    183.    214.    229=    229.
CRD$GL_HS_CURRENT_STATE  External Lib03   220=    309=    619=
CRD$GT_DS_COMMAND_OUT  External Lib03   875a
CRD$GT_DS_LOAD_COM  External Lib03   370m    379.
CRD$GT_DS_SELECT_COM  External Lib03   517m    529.
CRD$GT_DS_SET_EVENT_COM  External Lib03   465m    477.
CRD$GT_DS_SET_FLAGS_COM  External Lib03   416m    428.
CRD$GT_DS_START_COM  External Lib03   567m    576.
CRD$GT_FAIL   External Lib03   736a
CRD$GT_PASS   External Lib03   207a
CRD$INIT_DDB_STATUS  ExtRout  Lib03   150c
CRD$K_ACTION_EQL_DO_NOTHING Literal     93
CRD$K_ACTION_RANGE_ERROR Literal     93
CRD$K_ADVANCE_DIAGNOSTIC Literal     93
CRD$K_ADVANCE_GENERAL_COM Literal     93
CRD$K_ADVANCE_STATE  Literal     93
CRD$K_ALLOCATION_ERROR  Literal     93
CRD$K_ASSIGN_PTABLE_PTRS Literal     93
CRD$K_AUTOSIZER_ERROR  Literal     93
CRD$K_AUTOSIZER_SET_FLAGS Literal     93
CRD$K_BAD_ACTION_NUMBER  Literal     93
CRD$K_BAD_TYPECODE_ERROR Literal     93
CRD$K_BASE_SYSTEM_IDC  Literal     93
CRD$K_BASE_SYSTEM_LESI  Literal     93
CRD$K_BASE_SYSTEM_NULL  Literal     93
CRD$K_BASE_SYSTEM_UDA_0  Literal     93
CRD$K_BASE_SYSTEM_UDA_1  Literal     93
CRD$K_BASE_SYSTEM_UDA_2  Literal     93
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_BASE_SYSTEM_UDA_3  Literal     93
CRD$K_CRD_INITIALIZATION Literal     93
CRD$K_CREATE_HST  Literal     93
CRD$K_DATA_LENGTH_ERROR  Literal     93
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     93
CRD$K_DDB_BLINK   Literal     93
CRD$K_DDB_FLINK   Literal     93
CRD$K_DDB_LAST_ENTRY  Literal     93
CRD$K_DDB_MEMORY_SIZE  Literal     93
CRD$K_DDB_MENU_DEVICE_NUMB Literal     93
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     93   190
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     93
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     93   214
CRD$K_DDB_PTABLE_ENTRY  Literal     93   522
CRD$K_DDB_STATUS  Literal     93   206     209     210     211     243     244     249     656     690     728
            729     730     801     802     803
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     93
CRD$K_DEVICE_NAME  Literal     93
CRD$K_DIAGNOSTIC_ERROR  Literal     93
CRD$K_DIAGNOSTIC_NAME  Literal     93   376
CRD$K_DONE_GENERAL_COMS  Literal     93
CRD$K_DO_NOTHING  Literal     93
CRD$K_DS_COMMAND_SIZE  Literal  Lib03   387     436     485     538     584
CRD$K_DUMMY_10   Literal     93
CRD$K_DUMMY_11   Literal     93
CRD$K_DUMMY_12   Literal     93
CRD$K_DUMMY_13   Literal     93
CRD$K_DUMMY_3   Literal     93
CRD$K_DUMMY_4   Literal     93
CRD$K_DUMMY_5   Literal     93
CRD$K_DUMMY_6   Literal     93
CRD$K_DUMMY_7   Literal     93
CRD$K_DUMMY_8   Literal     93
CRD$K_DUMMY_9   Literal     93
CRD$K_EXIT_CRD   Literal     93
CRD$K_HOOK_POINT  Literal     93
CRD$K_HS_INTERNAL_ERROR  Literal     93
CRD$K_IDC_EVDLB   Literal     93
CRD$K_IDC_EVDLC   Literal     93
CRD$K_IDC_EVDLD   Literal     93
CRD$K_IDC_EVRAD_0  Literal     93
CRD$K_IDC_EVRAD_1  Literal     93
CRD$K_IDC_LAST_DIAGNOSTIC Literal     93
CRD$K_INTERNAL_ERROR  Literal     93
CRD$K_LAST_ACTION_ROUTINE Literal     93
CRD$K_LAST_ENTRY  Literal     93
CRD$K_LAST_FUNCTION  Literal     93
CRD$K_LAST_HOOK_POINT  Literal     93
CRD$K_LAST_INTERNAL_ERROR Literal     93
CRD$K_LAST_TYPE   Literal     93
CRD$K_LESI_ENWCA  Literal     93
CRD$K_LESI_EVRMB_0  Literal     93
CRD$K_LESI_EVRMB_1  Literal     93
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
CRD$K_LESI_LAST_DIAGNOSTIC Literal     93
CRD$K_LEVEL_4_PASSED  Literal      93
CRD$K_LOAD_AUTOSIZER  Literal      93
CRD$K_LOAD_ERROR  Literal      93
CRD$K_LOAD_GENERAL_COM  Literal      93
CRD$K_LOAD_LOAD_COM  Literal      93
CRD$K_LOAD_SELECT_COM  Literal      93
CRD$K_LOAD_SET_EVENT_COM Literal      93
CRD$K_LOAD_SET_FLAGS_COM Literal      93
CRD$K_LOAD_START_COM  Literal      93
CRD$K_LOAD_SUP_FILE  Literal      93
CRD$K_MM_INTERNAL_ERROR  Literal      93
CRD$K_NEW_LINE   Literal      93
CRD$K_NUMB_GENERAL_COMMANDS Literal  Lib03   308
CRD$K_NUMB_SUPPORT_ENTRIES Literal  Lib03   199
CRD$K_PREPARATION_ERROR  Literal      93
CRD$K_PREPARATION_ERROR_TEXT Literal      93
CRD$K_REPEAT_TURING  Literal  Lib03   152      252      313      425      474      625      813
CRD$K_RETURN_FAILURE  Literal  Lib03   843
CRD$K_RETURN_SUCCESS  Literal  Lib03   758
CRD$K_RUNTIME   Literal      93
CRD$K_SAME_LINE   Literal      93
CRD$K_SET_EVENT_ARGS  Literal      93   471
CRD$K_SET_FLAGS_ARGS  Literal      93   422
CRD$K_SET_UP_DIAGNOSTIC  Literal      93
CRD$K_START   Literal      93
CRD$K_START_AUTOSIZER  Literal      93
CRD$K_START_ERROR  Literal      93
CRD$K_START_QUALIFIERS  Literal      93   573
CRD$K_STAR_LINE   Literal      93
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal      93
CRD$K_STATE_ADVANCE_GENERAL_COM Literal      93
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal      93
CRD$K_STATE_AUTOSIZER_ERROR Literal      93
CRD$K_STATE_AUTOSIZER_RUNNING Literal      93
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal      93
CRD$K_STATE_DIAGNOSTIC_ERROR Literal      93
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal      93
CRD$K_STATE_DONE_DIAGNOSTICS Literal      93
CRD$K_STATE_DONE_GENERAL_COMS Literal      93
CRD$K_STATE_DUMMY_1  Literal      93
CRD$K_STATE_DUMMY_10  Literal      93
CRD$K_STATE_DUMMY_12  Literal      93
CRD$K_STATE_DUMMY_13  Literal      93
CRD$K_STATE_DUMMY_2  Literal      93
CRD$K_STATE_DUMMY_3  Literal      93
CRD$K_STATE_DUMMY_4  Literal      93
CRD$K_STATE_DUMMY_6  Literal      93
CRD$K_STATE_DUMMY_7  Literal      93
CRD$K_STATE_DUMMY_8  Literal      93
CRD$K_STATE_EXIT_CRD  Literal      93
CRD$K_STATE_INTERNAL_ERROR Literal      93
CRD$K_STATE_LAST_STATE  Literal      93
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_LEVEL_4_PASSED Literal     93
CRD$K_STATE_LOAD_AUTOSIZER Literal     93
CRD$K_STATE_LOAD_ERROR  Literal     93
CRD$K_STATE_LOAD_GENERAL_COM Literal     93
CRD$K_STATE_LOAD_LOAD_COM Literal     93
CRD$K_STATE_LOAD_SELECT_COM Literal     93
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     93
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     93
CRD$K_STATE_LOAD_START_COM Literal     93
CRD$K_STATE_PREPARATION_ERROR Literal     93
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     93
CRD$K_STATE_START  Literal     93
CRD$K_STATE_START_AUTOSIZER Literal     93
CRD$K_STATE_START_ERROR  Literal     93
CRD$K_SUCCESS   Unbound          387     436     485     538     584
    Literal  Lib03   387     436     485     538     584
CRD$K_TEST_START_TEXT  Literal     93
CRD$K_TQ_INTERNAL_ERROR  Literal     93   841
CRD$K_TYPECODE   Literal     93
CRD$K_UDA_0_EVDLB  Literal     93
CRD$K_UDA_0_EVDLC  Literal     93
CRD$K_UDA_0_EVDLD  Literal     93
CRD$K_UDA_0_EVRLA_0  Literal     93
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     93
CRD$K_UDA_1_EVDLB  Literal     93
CRD$K_UDA_1_EVDLC  Literal     93
CRD$K_UDA_1_EVDLD  Literal     93
CRD$K_UDA_1_EVRLA_0  Literal     93
CRD$K_UDA_1_EVRLA_1  Literal     93
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     93
CRD$K_UDA_2_EVDLB  Literal     93
CRD$K_UDA_2_EVDLC  Literal     93
CRD$K_UDA_2_EVDLD  Literal     93
CRD$K_UDA_2_EVRLA_0  Literal     93
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     93
CRD$K_UDA_3_EVDLB  Literal     93
CRD$K_UDA_3_EVDLC  Literal     93
CRD$K_UDA_3_EVDLD  Literal     93
CRD$K_UDA_3_EVRLA_0  Literal     93
CRD$K_UDA_3_EVRLA_1  Literal     93
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     93
CRD$LOAD_ERROR   ExtRout  Lib03   659c
CRD$LOAD_GENERAL_COM  ExtRout  Lib03   280c
CRD$PRINT   External Lib03   207ac   249ac   736ac   745ac   875ac
CRD$PRINT_DS_COMMAND  GlobRout   847 Lib03e     89f     385c     434c     483c     536c     582c
CRD$START_ERROR   ExtRout  Lib03   693c
CRD$STOP   ExtRout  Lib03   842c
DDB$V_STATUS_DEVICE_BAD  Macro    Lib03   209     210     728
DDB$V_STATUS_DEV_TEST_COMPLETE Macro    Lib03   730     801
DDB$V_STATUS_DIAGNOSTIC_ERROR Macro    Lib03   211     243
DDB$V_STATUS_MESSAGE_PRINTED Macro    Lib03   206     244     656     690     729     802
DDB$V_STATUS_PREPARATION_ERROR Macro    Lib03   803
DEVICE_DATA_BLOCK_STRUCTURE Structur Lib03   137     181     514     613     654     688     722     799
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DEVICE_NAME_ASCIC_PTR  Local      520   522=    526m    530a.
DIAG_NAME   Bind       376   380.
DS$ABORT   ExtRout    752   752c    811e    811c
DS$K_PRINTB   Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_PRINTF   Literal    207   207
    Literal    249   249
    Literal    736   736
    Literal    745   745
    Literal    875   875
DS$K_PRINTI   Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_PRINTX   Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ABORT_PROGRAM  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ABORT_TEST  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_COMMAND_ERR  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_COMMAND_OUT  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_CRD_AUTOTEST  Literal    207   207
    Literal    249   249
    Literal    736   736
    Literal    745   745
    Literal    875   875
DS$K_TYPE_DS_PROMPT  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DS$K_TYPE_DS_START  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ERRDEV  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ERRHARD  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ERROR_BODY  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ERROR_END  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ERRPREP  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ERRSOFT  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ERRSUP  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ERRSYS  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_ERR_HALT  Literal    207
    Literal    249
    Literal    736
    Literal    745
    Literal    875
DS$K_TYPE_EXCEPTION  Literal    207
    Literal    249
    Literal    736
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     745
    Literal     875
DS$K_TYPE_EXCEPTION_HEAD Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_FIRST_PASS  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_GENERAL  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_GENERAL_ERROR  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_NO_TESTS  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_PARAM_ERROR  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_PROGRAM_END  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_PROGRAM_INFO  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_PROGRAM_START  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_QIO_INVADP  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_QIO_NODRIVER  Literal     207
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_QIO_WRONGVER  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_SCRIPT_ECHO  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_SCRIPT_PNF  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_SCRIPT_PROMPT  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_SCRIPT_SKIP  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_SEQUENCE_ERROR Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_START_ERR  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_START_LIST  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_SUMMARY  Literal     207
    Literal     249
    Literal     736
    Literal     745
    Literal     875
DS$K_TYPE_USER_PROMPT  Literal     207
    Literal     249
    Literal     736
    Literal     745
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    875
DSA$AL_APTMAIL   Bind         92    92
DSA$GL_APTCOM    Bind         92
DSA$GL_DEVLEN    Bind         92
DSA$GL_DEVNAM    Bind         92
DSA$GL_ERRNO   Bind         92
DSA$GL_EVENT   Bind         92
DSA$GL_FLAGS   Bind         92   738     873
DSA$GL_MSGPTR    Bind         92
DSA$GL_MSGTYP    Bind         92
DSA$GL_PASSES    Bind         92
DSA$GL_PASSNO    Bind         92
DSA$GL_SECTNO    Bind         92
DSA$GL_SID   Bind         92
DSA$GL_SUBTNO    Bind         92
DSA$GL_TESTNO    Bind         92
DSA$GL_UNITS   Bind         92
DSA$V_CRD_TRACE    Macro    Lib02   738     873
ERROR_MESSAGE_ADDRESS  RoutForm   697   741.
ERROR_MESSAGE_LENGTH  RoutForm   697   742.
ERROR_TEXT_LEN   Bind         742   745a
ERROR_TEXT_PTR   Bind         741   745a
FALSE    Literal  Lib03   243     244     730     801
GET1ST_    Macro    Lib04    93     207     249     736     745     875
GET2ND_    Unbound            93     207     249     736     745     875
HARDWARE_SUPPORT_VECTOR  Structur Lib03   373     419     468     570
HS$ADVANCE_DIAGNOSTIC  GlobRout   156 Lib03e     69f
HS$ADVANCE_GENERAL_COM  GlobRout   284 Lib03e     72f
HS$CHANGE_TABLE   GlobRout   588 Lib03e     81f
HS$DIAGNOSTIC_ERROR  GlobRout   697 Lib03e     85f
HS$DONE_GENERAL_COMS  GlobRout   317 Lib03e     73f
HS$INIT_HS   GlobRout   112 Lib03e     68f
HS$INTERNAL_ERROR  GlobRout   817 Lib03e     87f
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal     93
HS$K_ACTION_ADVANCE_GENERAL_COM Literal     93
HS$K_ACTION_ADVANCE_STATE Literal     93
HS$K_ACTION_CHANGE_TABLE Literal     93
HS$K_ACTION_DIAGNOSTIC_ERROR Literal     93
HS$K_ACTION_DONE_GENERAL_COMS Literal     93
HS$K_ACTION_DO_NOTHING  Literal     93
HS$K_ACTION_DUMMY_3  Literal     93
HS$K_ACTION_DUMMY_4  Literal     93
HS$K_ACTION_DUMMY_5  Literal     93
HS$K_ACTION_DUMMY_6  Literal     93
HS$K_ACTION_INIT_HS  Literal     93
HS$K_ACTION_INTERNAL_ERROR Literal     93
HS$K_ACTION_LAST_ACTION  Literal     93
HS$K_ACTION_LOAD_ERROR  Literal     93
HS$K_ACTION_LOAD_GENERAL_COM Literal     93
HS$K_ACTION_LOAD_LOAD_COM Literal     93
HS$K_ACTION_LOAD_SELECT_COM Literal     93
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     93
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     93
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_ACTION_LOAD_START_COM Literal      93
HS$K_ACTION_PREPARATION_ERROR Literal      93
HS$K_ACTION_START_ERROR  Literal      93
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal      93
HS$K_STATE_ADVANCE_GENERAL_COM Literal      93
HS$K_STATE_CHANGE_TABLE  Literal      93   220
HS$K_STATE_DIAGNOSTIC_ERROR Literal      93
HS$K_STATE_DIAGNOSTIC_RUNNING Literal      93
HS$K_STATE_DONE_GENERAL_COMS Literal      93   309
HS$K_STATE_DUMMY_1  Literal      93
HS$K_STATE_DUMMY_2  Literal      93
HS$K_STATE_DUMMY_3  Literal      93
HS$K_STATE_DUMMY_4  Literal      93
HS$K_STATE_DUMMY_6  Literal      93
HS$K_STATE_INIT_HS  Literal      93   619
HS$K_STATE_INTERNAL_ERROR Literal      93
HS$K_STATE_LAST_STATE  Literal      93
HS$K_STATE_LOAD_ERROR  Literal      93
HS$K_STATE_LOAD_GENERAL_COM Literal      93
HS$K_STATE_LOAD_LOAD_COM Literal      93
HS$K_STATE_LOAD_SELECT_COM Literal      93
HS$K_STATE_LOAD_SET_EVENT_COM Literal      93
HS$K_STATE_LOAD_SET_FLAGS_COM Literal      93
HS$K_STATE_LOAD_START_COM Literal      93
HS$K_STATE_PREPARATION_ERROR Literal      93
HS$K_STATE_START_ERROR  Literal      93
HS$LOAD_ERROR   GlobRout   629 Lib03e      83f
HS$LOAD_GENERAL_COM  GlobRout   256 Lib03e      71f
HS$LOAD_LOAD_COM  GlobRout   345 Lib03e      75f
HS$LOAD_SELECT_COM  GlobRout   489 Lib03e      78f
HS$LOAD_SET_EVENT_COM  GlobRout   440 Lib03e      77f
HS$LOAD_SET_FLAGS_COM  GlobRout   391 Lib03e      76f
HS$LOAD_START_COM  GlobRout   542 Lib03e      79f
HS$PREPARATION_ERROR  GlobRout   762 Lib03e      86f
HS$START_ERROR   GlobRout   663 Lib03e      84f
LENGTH    RoutForm   817   841.
MEN$FNCTST_TESTSTRT_TEXT ExtRout  Lib03   235c
MEN$K_HS_STATE_TABLE  Literal      93
MEN$K_MM_STATE_TABLE  Literal      93
MEN$K_TQ_STATE_TABLE  Literal      93   622
MEN$MENU_TEST_INTERNAL_ERROR ExtRout  Lib03   841c
MEN$PREPARATION_ERR_TEXT ExtRout  Lib03   809c
MENU$K_EXIT_AUTOSIZER_ERROR Literal      93
MENU$K_EXIT_INTERNAL_ERROR Literal      93   842
MENU$K_EXIT_LAST_REASON  Literal      93
MENU$K_EXIT_OPERATOR_ABORT Literal      93
MENU$K_EXIT_OPERATOR_STOP Literal      93
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal      93
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal      93
MM$K_ACTION_ADVANCE_STATE Literal      93
MM$K_ACTION_AUTOSIZER_ERROR Literal      93
MM$K_ACTION_BUILD_DDBS  Literal      93
MM$K_ACTION_BUILD_HSTS  Literal      93
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_ACTION_CHANGE_TABLE Literal     93
MM$K_ACTION_DONE_INITS  Literal      93
MM$K_ACTION_DO_NOTHING  Literal      93
MM$K_ACTION_DUMMY_3  Literal     93
MM$K_ACTION_DUMMY_4  Literal     93
MM$K_ACTION_DUMMY_5  Literal     93
MM$K_ACTION_DUMMY_6  Literal     93
MM$K_ACTION_DUMMY_7  Literal     93
MM$K_ACTION_DUMMY_8  Literal     93
MM$K_ACTION_INTERNAL_ERROR Literal      93
MM$K_ACTION_LAST_ACTION  Literal     93
MM$K_ACTION_LOAD_AUTOSIZER Literal      93
MM$K_ACTION_MENU_PROMPT  Literal     93
MM$K_ACTION_PRINT_MENU  Literal      93
MM$K_ACTION_PRINT_MENU_HEADING Literal      93
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal      93
MM$K_ACTION_START_AUTOSIZER Literal      93
MM$K_STATE_AUTOSIZER_ERROR Literal      93
MM$K_STATE_AUTOSIZER_RUNNING Literal      93
MM$K_STATE_BUILD_DDBS  Literal      93
MM$K_STATE_BUILD_HSTS  Literal      93
MM$K_STATE_CHANGE_TABLE  Literal     93
MM$K_STATE_DONE_INITS  Literal      93
MM$K_STATE_DUMMY_1  Literal     93
MM$K_STATE_DUMMY_2  Literal     93
MM$K_STATE_DUMMY_3  Literal     93
MM$K_STATE_DUMMY_5  Literal     93
MM$K_STATE_DUMMY_7  Literal     93
MM$K_STATE_DUMMY_8  Literal     93
MM$K_STATE_INTERNAL_ERROR Literal      93
MM$K_STATE_LAST_STATE  Literal      93
MM$K_STATE_LOAD_AUTOSIZER Literal      93
MM$K_STATE_MENU_PROMPT  Literal      93
MM$K_STATE_PRINT_MENU  Literal      93
MM$K_STATE_PRINT_MENU_HEADING Literal      93
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal      93
MM$K_STATE_START  Literal      93
MM$K_STATE_START_AUTOSIZER Literal      93
MODNAM    Macro    Lib01   109
NUL2ND    Macro    Lib04    93     207     249     736     745     875
SET_EVENT_ARGS   Bind        471   473.    478.
SET_FLAGS_ARGS   Bind        422   424.    429.
START_QUALIFIERS  Bind        573   577.
TQ$K_ACTION_ADVANCE_STATE Literal      93
TQ$K_ACTION_CHANGE_TABLE Literal      93
TQ$K_ACTION_DONE_TEST_QUEUE Literal      93
TQ$K_ACTION_DO_NOTHING  Literal      93
TQ$K_ACTION_DUMMY_3  Literal     93
TQ$K_ACTION_DUMMY_4  Literal     93
TQ$K_ACTION_DUMMY_5  Literal     93
TQ$K_ACTION_GET_DDB  Literal     93
TQ$K_ACTION_INIT_TQ  Literal     93
TQ$K_ACTION_INTERNAL_ERROR Literal      93
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- --------  ------
TQ$K_ACTION_LAST_ACTION  Literal     93
TQ$K_STATE_CHANGE_TABLE  Literal     93
TQ$K_STATE_DONE_TEST_QUEUE Literal     93
TQ$K_STATE_DUMMY_1  Literal     93
TQ$K_STATE_DUMMY_2  Literal     93
TQ$K_STATE_DUMMY_3  Literal     93
TQ$K_STATE_DUMMY_4  Literal     93
TQ$K_STATE_DUMMY_5  Literal     93
TQ$K_STATE_GET_DDB  Literal     93
TQ$K_STATE_INIT_TQ  Literal     93
TQ$K_STATE_INTERNAL_ERROR Literal     93
TQ$K_STATE_LAST_STATE  Literal     93
TRUE    Literal  Lib03   656     690     728     729     802     803
TYPECODE   RoutForm   697
    RoutForm   817   841.


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDHSACT.B32;1
    2  Module CRDHSACT
   55  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   58  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   61  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   64  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  879  Eludom CRDHSACT


KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     @ Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;              671         1         0        46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
```

```
;            923          5          0         94          00:00.2
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;            486         64         13         62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3          0      1000          00:04.3




;     COMMAND QUALIFIERS

; BLISS CRDHSACT.B32/LIST=CRDHSACT.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDHSACT.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  925 code + 134 data bytes
; Run Time:    00:36.6
; Elapsed Time:    01:14.5
; Lines/CPU Min:    1439
; Lexemes/CPU-Min: 54995
; Memory Used: 220 pages
; Compilation Complete
```

```
;    0001   0 %TITLE '*** CRDINIT Module'
;    0002   0 MODULE CRDINIT (    ! The initialization routine(s) for CRD
;    0003   0   IDENT = '01-06'
;    0004   0   ) =
;    0005   0 !++
;    0006   0 ! COPYRIGHT (c) 1980, 1981
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 !-
;    0024   0 !++
;    0025   0 ! Facility:
;    0026   0 !
;    0027   0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028   0 !
;    0029   0 ! Abstract:
;    0030   0 !
;    0031   0 ! This module contains the routine(s) necessary to initialize CRD.
;    0032   0 !
;    0033   0 ! Author:
;    0034   0 !
;    0035   0 ! Jack Stansbury
;    0036   0 !
;    0037   0 ! Creation Date:
;    0038   0 !
;    0039   0 ! 1-April-1982
;    0040   0 !
;    0041   0 ! Version:
;    0042   0 !
;    0043   0 ! 6.9
;    0044   0 !
;    0045   0 ! Modified By:
;    0046   0 !
;    0047   0 ! 01 Jack Stansbury, Version 1.1, July 25, 1982
;    0048   0 !  Added the CRD$Common_Initializations routine. This contains code that
;    0049   0 !  will be excuted for both Menu Test and Auto Test. This routine is passed the address
;    0050   0 !  of the ^C handler routine, since Menu Test and Auto Test use different handler
;    0051   0 !  routines.
;    0052   0 !
;    0053   0 ! 02 Jack Stansbury, Version 1.1, July 25, 1982
;    0054   0 !  Changed the above routine to return a flag saying whether Menu or Auto Test was running.
;    0055   0 !  This is used by both the main Menu and main Auto initialization routines to set the
```

```
;    0056  0 !  appropriate DSA flag bit.
;    0057  0 !
;    0058  0 ! 03 Jack Stansbury, Version 1.1, July 25, 1982
;    0059  0 !  Added the CRD$GT_Which_CRD global string pointer. Also, changed what I did above since
;    0060  0 !  the only routines that will be calling the CRD$Common_Initializations routine is the two
;    0061  0 !  init routines for Menu and Auto. Since these two routines already know which DSA bit should
;    0062  0 !  be set, there is no need to save this information.
;    0063  0 !
;    0064  0 ! 04 Jack Stansbury, Version 1.1, July 27, 1982
;    0065  0 !  Changed the establishment of the VDS ^C handler routine.
;    0066  0 !
;    0067  0 ! 05 Jack Stansbury, Version 1.1, August 19, 1982
;    0068  0 !  Took out the CRD$GB_CRD_Debug variable, because it is in the CRDVECS module now.
;    0069  0 !
;    0070  0 ! 06 John Ciukaj Version 1.2 4-Apr-1982
;    0071  0 !  - Changed references to CRD bits in DSA Flags to
;    0072  0 !    distinguish between Offline and online
;    0073  0 !  - In CRD$CRD_Initialization do not invoke CRD$Build_DDBs ();.
;    0074  0 !    Must wait for Autosizer to be run in order to determine
;    0075  0 !    which Hardware Support Table to use.
;    0076  0 !
;    0077  0 !
;    0078  0 !--
;    0079  0 %SBTTL 'Libraries'
;    0080  1 BEGIN     ! Begin the CRDINIT module
;    0081  1
;    0082  1 LIBRARY
;    0083  1  '$CRD';
;    0084  1
;    0085  1 LIBRARY
;    0086  1  '$DS';
;    0087  1
;    0088  1 LIBRARY
;    0089  1  '$Diag';
;    0090  1
;    0091  1 LIBRARY
;    0092  1  'Sys$Library:Lib';
;    0093  1 %SBTTL 'Table of Contents'
;    0094  1
;    0095  1 FORWARD ROUTINE
;    0096  1  CRD$Common_Initializations : ADDRESSING_MODE (LONG_RELATIVE),     ![01]
;    0097  1       ! The common initializations for Menu and Auto Test.   ![01]
;    0098  1
;    0099  1  CRD$CRD_Initialization  : ADDRESSING_MODE (LONG_RELATIVE);
;    0100  1       ! The initialization routine for the CRD Auto Test code.
;    0101  1 %SBTTL 'Included Macros and Literals'
;    0102  1
;    0103  1  $CRD_Literals;   ! Define some literals for CRD
;    0104  1  $DS_DSADef;   ! Define the DSA flag bits
;    0105  1 %SBTTL 'Psect Definitions'
;    0106  1
;    0107  1 PSECT
;    0108  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0109  1
;    0110  1 PSECT
```

```
;    0111  1      CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0112  1
;    0113  1 PSECT
;    0114  1      OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0115  1
;    0116  1 PSECT
;    0117  1      GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0118  1 %SBTTL 'Module-Global Static Storage'
;    0119  1
;    0120  1  ModNam ('CRDINIT');  ! Module name for ErrSup macro
;    0121  1 %SBTTL 'Global Storage'
;    0122  1
;    0123  1 GLOBAL
;    0124  1  CRD$GT_Which_CRD,  ! Points to a string containing either 'MENU" or "AUTO'.  ![03]
;    0125  1
;    0126  1  CRD$GL_Saved_DSA_Flags;  ! To save the settings of the DSA flags
;    0127  1 %SBTTL 'CRD$CRD_Initialization Routine'
;    0128  1
;    0129  1 GLOBAL ROUTINE CRD$CRD_Initialization =
;    0130  1
;    0131  1 !++
;    0132  1 ! Functional Description:
;    0133  1 !
;    0134  1 ! This routine is the main initialization routine for the CRD Auto Test code. This routine is NOT called  ![03]
;    0135  1 ! from the Menu Test routines!!!!!!          ![03]
;    0136  1 !
;    0137  1 ! Formal Parameters:
;    0138  1 !
;    0139  1 ! None
;    0140  1 !
;    0141  1 ! Side Effects:
;    0142  1 !
;    0143  1 ! Plenty
;    0144  1 !
;    0145  1 ! Completion Codes:
;    0146  1 !
;    0147  1 ! 1 => The initialization succeeded
;    0148  1 ! 0 => The initialization failed
;    0149  1 !--
;    0150  2 BEGIN     ! Routine CRD$CRD_Initialization
;    0151  2  CRD$Common_Initializations (CRD$Control_C_Handler);       ![03]
;    0152  2      ! Perform common ones first      ![01]
;    0153  2
;    0154  2  CRD$Init_General_Com_Table (); ! Initialize the General_Commands table and data structures
;    0155  2
;    0156  2  IF .CRD$GB_CRD_Debug THEN
;    0157  2   CRD$Print_Command_Table ();
;    0158  2      ! Print the contents of the general command table
;    0159  2
;    0160  2  IF .CRD$GB_CRD_Debug THEN
;    0161  2   CRD$Print_PTables (); ! Print the contents of the PTables
;    0162  2
;    0163  2  CRD$Init State_Table (); ! Initialize the State_Table
;    0164  2
;    0165  2  CRD$Build_Support_Table (); ! Build the Hardware Support table
```

```
;    0166  2
;    0167  2
;    0168  2  IF .CRD$GB_CRD_Debug THEN
;    0169  2   CRD$Print_Hardware_Support ();
;    0170  2      ! Print the Hardware Support table
;    0171  2
;    0172  2  DSA$V_Binary = On;  ! Turn these two back on now after the initialization is all done.
;    0173  2  DSA$V_CRD_AutoTest_Off = On; ! Leave this on for the duration of the CRD Auto Test execution. ![06]
;    0174  2
;    0175  3  RETURN (CRD$K_Success)
;    0176  1 END;     ! Routine CRD$CRD_Initialization


            .TITLE  CRDINIT *** CRDINIT Module
            .IDENT  \01-06\

            .PSECT  WORK,NOEXE,2

            00000 CRD$GT_WHICH_CRD::
            .BLKB   4
            00004 CRD$GL_SAVED_DSA_FLAGS::
            .BLKB   4

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

        54  49  4E  49  44  52  43  07  00000 P.AAA:  .ASCII  <7>\CRDINIT\          ;

           DSA$AL_APTMAIL=     65024
           DSA$GL_FLAGS=       65024
           DSA$GL_APTCOM=      65028
           DSA$GL_PASSES=      65032
           DSA$GL_UNITS=       65036
           DSA$GL_SECTNO=      65040
           DSA$GL_SID=       65044
           DSA$GL_MSGTYP=      65088
           DSA$GL_ERRNO=       65092
           DSA$GL_EVENT=       65096
           DSA$GL_SUBTNO=      65100
           DSA$GL_TESTNO=      65104
           DSA$GL_PASSNO=      65108
           DSA$GL_DEVLEN=      65112
           DSA$GL_DEVNAM=      65116
           DSA$GL_MSGPTR=      65128
           $MODULE=        P.AAA
            .EXTRN  CRD$COMMON_INITIALIZATIONS
            .EXTRN  CRD$CRD_INITIALIZATION
            .EXTRN  CRD$CONTROL_C_HANDLER
            .EXTRN  CRD$INIT_GENERAL_COM_TABLE
            .EXTRN  CRD$GB_CRD_DEBUG
            .EXTRN  CRD$PRINT_COMMAND_TABLE
            .EXTRN  CRD$PRINT_PTABLES
            .EXTRN  CRD$INIT_STATE_TABLE
            .EXTRN  CRD$BUILD_SUPPORT_TABLE
            .EXTRN  CRD$PRINT_HARDWARE_SUPPORT
```

```
                .PSECT  CODE,NOWRT,  SHR,  PIC,2

                0000 00000      .ENTRY  CRD$CRD_INITIALIZATION, Save nothing       ; 0129
        00000000G   EF  9F 00002      PUSHAB  CRD$CONTROL_C_HANDLER             ; 0151
        00000000V   EF    01  FB 00008      CALLS   #1, CRD$COMMON_INITIALIZATIONS       ;
        00000000G   EF    00  FB 0000F      CALLS   #0, CRD$INIT_GENERAL_COM_TABLE       ; 0154
            15 00000000G    EF  E9 00016      BLBC    CRD$GB_CRD_DEBUG, 1$          ; 0156
        00000000G   EF    00  FB 0001D      CALLS   #0, CRD$PRINT_COMMAND_TABLE         ; 0157
            07 00000000G    EF  E9 00024      BLBC    CRD$GB_CRD_DEBUG, 1$          ; 0160
        00000000G   EF    00  FB 0002B      CALLS   #0, CRD$PRINT_PTABLES               ; 0161
        00000000G   EF    00  FB 00032 1$:    CALLS   #0, CRD$INIT_STATE_TABLE          ; 0163
        00000000G   EF    00  FB 00039      CALLS   #0, CRD$BUILD_SUPPORT_TABLE         ; 0165
            07 00000000G    EF  E9 00040      BLBC    CRD$GB_CRD_DEBUG, 2$          ; 0168
        00000000G   EF    00  FB 00047      CALLS   #0, CRD$PRINT_HARDWARE_SUPPORT      ; 0169
        0000FE00    9F    0802    8F  A8 0004E 2$:    BISW2   #2050, a#^X0000FE00         ; 0173
            50      01  D0 00057     MOVL    #1, R0              ; 0175
             04 0005A     RET                 ; 0176

; Routine Size:  91 bytes,    Routine Base:  CODE + 0000


;   0177  1 %SBTTL 'CRD$Common_Initializations Routine'          ![01]
;   0178  1
;   0179  1 GLOBAL ROUTINE CRD$Common_Initializations (Control_C_Handler_Address) =      ![01]
;   0180  1
;   0181  1 !++                 !etc.
;   0182  1 ! Functional Description:
;   0183  1 !
;   0184  1 ! This routine does any initilizations that are common to both the Menu Test and the Auto Test.
;   0185  1 ! This is called by both of the main initialization routines for Menu and Auto test.   ![03]
;   0186  1 !
;   0187  1 ! Formal Parameters:
;   0188  1 !
;   0189  1 ! Control_C_Handler_Address: The address of the ^C handler routine.
;   0190  1 !
;   0191  1 ! Side Effects:
;   0192  1 !
;   0193  1 ! Plenty
;   0194  1 !
;   0195  1 ! Completion Codes:
;   0196  1 !
;   0197  1 ! CRD$K_Success
;   0198  1 !--
;   0199  2 BEGIN     ! Routine CRD$Common_Initializations
;   0200  2  LOCAL
;   0201  2   DS_Flags;  ! Used locally to set bits in the DS$GL_Flags longword
;   0202  2
;   0203  2  IF (.DSA$V_CRD_AutoTest_Off) THEN          ![06]
;   0204  2   CRD$GT_Which_CRD = CRD$GT_Auto         ![02]
;   0205  2  ELSE                ![02]
;   0206  2   CRD$GT_Which_CRD = CRD$GT_Menu;          ![02]
;   0207  2
;   0208  2  CRD$Set_Ctrl_C_Handlers (.Control_C_Handler_Address, .Control_C_Handler_Address);   ![04]
;   0209  2  CRD$Enable_Ctrl_C ();           ![04]
;   0210  2
```

```
;    0211  2  DSA$V_Binary     = Off; ! Turn it off before saving the flags
;    0212  2  DSA$V_CRD_AutoTest_Off = Off; ! Turn it off before saving the flags     [06]
;    0213  2  DSA$V_CRD_MenuTest_Off = Off; ! Turn it off before saving the flags     [06]
;    0214  2  DSA$V_CRD_MenuTest_On = Off; ! Turn it off before saving the flags     [06]
;    0215  2
;    0216  2  CRD$GL_Saved_DSA_Flags = .DSA$GL_Flags;
;    0217  2      ! Save the current settings of the DSA flags
;    0218  2
;    0219  2  DSA$V_APT     = Off; ! None of these DSA flags should be on
;    0220  2  DSA$V_Bell     = Off;
;    0221  2  DSA$V_Halt     = Off;
;    0222  2  DSA$V_IE1     = Off;
;    0223  2  DSA$V_IE2     = Off;
;    0224  2  DSA$V_IE3     = Off;
;    0225  2  DSA$V_IES     = Off;
;    0226  2  DSA$V_Loop     = Off;
;    0227  2  DSA$V_NoRpt     = Off;
;    0228  2  DSA$V_Oper     = On; ! Kludge
;    0229  2  DSA$V_Prompt     = Off;
;    0230  2  DSA$V_QA     = Off;
;    0231  2  DSA$V_Quick     = Off;
;    0232  2  DSA$V_Search     = Off;
;    0233  2  DSA$V_Trace     = Off;
;    0234  2  DSA$V_Verify     = Off;
;    0235  2
;    0236  3  RETURN (CRD$K_Success)           ![03]
;    0237  1 END;     ! Routine CRD$Common_Initializations     ![01]



        .EXTRN  CRD$GT_AUTO, CRD$GT_MENU
        .EXTRN  CRD$SET_CTRL_C_HANDLERS
        .EXTRN  CRD$ENABLE_CTRL_C

          0FFC 00000     .ENTRY  CRD$COMMON_INITIALIZATIONS, Save R2,R3,R4,- ; 0179
         R5,R6,R7,R8,R9,R10,R11      ;
         0D 0000FE01   9F      03  E1 00002    BBC     #3, @#^X0000FE01, 1$          ; 0203
      00000000'  EF 00000000G  EF  9E 0000A    MOVAB   CRD$GT_AUTO, CRD$GT_WHICH_CRD        ; 0204
         0B  11 00015    BRB     2$          ;
      00000000'  EF 00000000G  EF  9E 00017 1$:   MOVAB   CRD$GT_MENU, CRD$GT_WHICH_CRD        ; 0206
         51 04   AC  D0 00022 2$:   MOVL    CONTROL_C_HANDLER_ADDRESS, R1        ; 0208
         50 04   AC  D0 00026    MOVL    CONTROL_C_HANDLER_ADDRESS, R0        ;
      00000000G  EF  16 0002A    JSB     CRD$SET_CTRL_C_HANDLERS        ;
      00000000G  EF  16 00030    JSB     CRD$ENABLE_CTRL_C          ; 0209
      0000FE00   9F 00050802   8F  CA 00036    BICL2   #329730, @#^X0000FE00          ; 0214
      00000000'  EF 0000FE00   9F  D0 00041    MOVL    @#^X0000FE00, CRD$GL_SAVED_DSA_FLAGS       ; 0216
      0000FE00   9F 880000FD   8F  CA 0004C    BICL2   #-2013265667, @#^X0000FE00        ; 0227
      0000FE01   9F       10  88 00057    BISB2   #16, @#^X0000FE01          ; 0228
      0000FE01   9F E7    8F  8A 0005E    BICB2   #231, @#^X0000FE01          ; 0234
         50       01  D0 00066    MOVL    #1, R0          ; 0236
       04 00069    RET                 ; 0237

; Routine Size:  106 bytes,    Routine Base:  CODE + 005B
```

```
;   0238  1 END     ! End of CRDINIT module
;   0239  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA             8  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK             8  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE           197  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
Symbol                            Type  Defined  Referenced ...
--------------------------------- -------- ----- ---------------
$CRD_LITERALS   Macro    Lib01   103
$DS_DSADEF   Macro    Lib03   104
$EQULST    Macro    Lib04   103
$ER    Comptime   120
$MODULE    Bind       120
AUTO$K_EXIT_AUTOSIZER_ERROR Literal    103
AUTO$K_EXIT_CONTROL_C  Literal    103
AUTO$K_EXIT_DUMMY_2  Literal    103
AUTO$K_EXIT_DUMMY_3  Literal    103
AUTO$K_EXIT_ERRORS_COMPLETE Literal    103
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal    103
AUTO$K_EXIT_INTERNAL_ERROR Literal    103
AUTO$K_EXIT_INV_BASE_SYSTEM Literal    103
AUTO$K_EXIT_LAST_REASON  Literal    103   103
AUTO$K_EXIT_NOERRORS_COMPLETE Literal    103
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal    103
AUTO$K_EXIT_NOERRORS_PREP Literal    103
CONTROL_C_HANDLER_ADDRESS RoutForm   179   208.
CRD$BUILD_SUPPORT_TABLE  ExtRout  Lib01   165c
CRD$COMMON_INITIALIZATIONS GlobRout   179 Lib01e     96f    151c
CRD$CONTROL_C_HANDLER  ExtRout  Lib01   151a
CRD$CRD_INITIALIZATION  GlobRout   129 Lib01e     99f
CRD$ENABLE_CTRL_C  ExtRout  Lib01   209c
CRD$GB_CRD_DEBUG  External Lib01   156.    160.    168.
CRD$GL_SAVED_DSA_FLAGS  Global     126 Lib01e    216=
CRD$GT_AUTO   External Lib01   204a
CRD$GT_MENU   External Lib01   206a
CRD$GT_WHICH_CRD  Global     124 Lib01e    204=    206=
CRD$INIT_GENERAL_COM_TABLE ExtRout  Lib01   154c
CRD$INIT_STATE_TABLE  ExtRout  Lib01   163c
CRD$K_ACTION_EQL_DO_NOTHING Literal    103
CRD$K_ACTION_RANGE_ERROR Literal    103
CRD$K_ADVANCE_DIAGNOSTIC Literal    103
```

Symbol	Type	Defined	Referenced ...
CRD$K_ADVANCE_GENERAL_COM	Literal	103	
CRD$K_ADVANCE_STATE	Literal	103	
CRD$K_ALLOCATION_ERROR	Literal	103	
CRD$K_ASSIGN_PTABLE_PTRS	Literal	103	
CRD$K_AUTOSIZER_ERROR	Literal	103	
CRD$K_AUTOSIZER_SET_FLAGS	Literal	103	
CRD$K_BAD_ACTION_NUMBER	Literal	103	
CRD$K_BAD_TYPECODE_ERROR	Literal	103	
CRD$K_BASE_SYSTEM_IDC	Literal	103	
CRD$K_BASE_SYSTEM_LESI	Literal	103	
CRD$K_BASE_SYSTEM_NULL	Literal	103	
CRD$K_BASE_SYSTEM_UDA_0	Literal	103	
CRD$K_BASE_SYSTEM_UDA_1	Literal	103	
CRD$K_BASE_SYSTEM_UDA_2	Literal	103	
CRD$K_BASE_SYSTEM_UDA_3	Literal	103	
CRD$K_CRD_INITIALIZATION	Literal	103	
CRD$K_CREATE_HST	Literal	103	
CRD$K_DATA_LENGTH_ERROR	Literal	103	
CRD$K_DDB_AUTO_SUPPORT_ENTRY	Literal	103	
CRD$K_DDB_BLINK	Literal	103	
CRD$K_DDB_FLINK	Literal	103	
CRD$K_DDB_LAST_ENTRY	Literal	103	
CRD$K_DDB_MEMORY_SIZE	Literal	103	
CRD$K_DDB_MENU_DEVICE_NUMB	Literal	103	
CRD$K_DDB_MENU_SUPPORT_ENTRY	Literal	103	
CRD$K_DDB_MENU_SUPPORT_SIZE	Literal	103	
CRD$K_DDB_NUMB_SUPPORT_TABLES	Literal	103	
CRD$K_DDB_PTABLE_ENTRY	Literal	103	
CRD$K_DDB_STATUS	Literal	103	
CRD$K_DDB_SUPBLK_BLOCK_PTR	Literal	103	
CRD$K_DEVICE_NAME	Literal	103	
CRD$K_DIAGNOSTIC_ERROR	Literal	103	
CRD$K_DIAGNOSTIC_NAME	Literal	103	
CRD$K_DONE_GENERAL_COMS	Literal	103	
CRD$K_DO_NOTHING	Literal	103	
CRD$K_DUMMY_10	Literal	103	
CRD$K_DUMMY_11	Literal	103	
CRD$K_DUMMY_12	Literal	103	
CRD$K_DUMMY_13	Literal	103	
CRD$K_DUMMY_3	Literal	103	
CRD$K_DUMMY_4	Literal	103	
CRD$K_DUMMY_5	Literal	103	
CRD$K_DUMMY_6	Literal	103	
CRD$K_DUMMY_7	Literal	103	
CRD$K_DUMMY_8	Literal	103	
CRD$K_DUMMY_9	Literal	103	
CRD$K_EXIT_CRD	Literal	103	
CRD$K_HOOK_POINT	Literal	103	
CRD$K_HS_INTERNAL_ERROR	Literal	103	
CRD$K_IDC_EVDLB	Literal	103	
CRD$K_IDC_EVDLC	Literal	103	
CRD$K_IDC_EVDLD	Literal	103	
CRD$K_IDC_EVRAD_0	Literal	103	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_IDC_EVRAD_1  Literal    103
CRD$K_IDC_LAST_DIAGNOSTIC Literal    103
CRD$K_INTERNAL_ERROR  Literal    103
CRD$K_LAST_ACTION_ROUTINE Literal    103
CRD$K_LAST_ENTRY  Literal    103
CRD$K_LAST_FUNCTION  Literal    103
CRD$K_LAST_HOOK_POINT  Literal    103
CRD$K_LAST_INTERNAL_ERROR Literal    103
CRD$K_LAST_TYPE   Literal    103
CRD$K_LESI_ENWCA  Literal    103
CRD$K_LESI_EVRMB_0  Literal    103
CRD$K_LESI_EVRMB_1  Literal    103
CRD$K_LESI_LAST_DIAGNOSTIC Literal    103
CRD$K_LEVEL_4_PASSED  Literal    103
CRD$K_LOAD_AUTOSIZER  Literal    103
CRD$K_LOAD_ERROR  Literal    103
CRD$K_LOAD_GENERAL_COM  Literal    103
CRD$K_LOAD_LOAD_COM  Literal    103
CRD$K_LOAD_SELECT_COM  Literal    103
CRD$K_LOAD_SET_EVENT_COM Literal    103
CRD$K_LOAD_SET_FLAGS_COM Literal    103
CRD$K_LOAD_START_COM  Literal    103
CRD$K_LOAD_SUP_FILE  Literal    103
CRD$K_MM_INTERNAL_ERROR  Literal    103
CRD$K_NEW_LINE   Literal    103
CRD$K_PREPARATION_ERROR  Literal    103
CRD$K_PREPARATION_ERROR_TEXT Literal    103
CRD$K_RUNTIME   Literal    103
CRD$K_SAME_LINE   Literal    103
CRD$K_SET_EVENT_ARGS  Literal    103
CRD$K_SET_FLAGS_ARGS  Literal    103
CRD$K_SET_UP_DIAGNOSTIC  Literal    103
CRD$K_START   Literal    103
CRD$K_START_AUTOSIZER  Literal    103
CRD$K_START_ERROR  Literal    103
CRD$K_START_QUALIFIERS  Literal    103
CRD$K_STAR_LINE   Literal    103
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal    103
CRD$K_STATE_ADVANCE_GENERAL_COM Literal    103
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal    103
CRD$K_STATE_AUTOSIZER_ERROR Literal    103
CRD$K_STATE_AUTOSIZER_RUNNING Literal    103
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal    103
CRD$K_STATE_DIAGNOSTIC_ERROR Literal    103
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal    103
CRD$K_STATE_DONE_DIAGNOSTICS Literal    103
CRD$K_STATE_DONE_GENERAL_COMS Literal    103
CRD$K_STATE_DUMMY_1  Literal    103
CRD$K_STATE_DUMMY_10  Literal    103
CRD$K_STATE_DUMMY_12  Literal    103
CRD$K_STATE_DUMMY_13  Literal    103
CRD$K_STATE_DUMMY_2  Literal    103
CRD$K_STATE_DUMMY_3  Literal    103
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_DUMMY_4  Literal     103
CRD$K_STATE_DUMMY_6  Literal     103
CRD$K_STATE_DUMMY_7  Literal     103
CRD$K_STATE_DUMMY_8  Literal     103
CRD$K_STATE_EXIT_CRD  Literal     103
CRD$K_STATE_INTERNAL_ERROR Literal     103
CRD$K_STATE_LAST_STATE  Literal     103
CRD$K_STATE_LEVEL_4_PASSED Literal     103
CRD$K_STATE_LOAD_AUTOSIZER Literal     103
CRD$K_STATE_LOAD_ERROR  Literal     103
CRD$K_STATE_LOAD_GENERAL_COM Literal     103
CRD$K_STATE_LOAD_LOAD_COM Literal     103
CRD$K_STATE_LOAD_SELECT_COM Literal     103
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     103
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     103
CRD$K_STATE_LOAD_START_COM Literal     103
CRD$K_STATE_PREPARATION_ERROR Literal     103
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     103
CRD$K_STATE_START  Literal     103
CRD$K_STATE_START_AUTOSIZER Literal     103
CRD$K_STATE_START_ERROR  Literal     103
CRD$K_SUCCESS   Literal  Lib01   175      236
CRD$K_TEST_START_TEXT  Literal     103
CRD$K_TQ_INTERNAL_ERROR  Literal     103
CRD$K_TYPECODE   Literal     103
CRD$K_UDA_0_EVDLB  Literal     103
CRD$K_UDA_0_EVDLC  Literal     103
CRD$K_UDA_0_EVDLD  Literal     103
CRD$K_UDA_0_EVRLA_0  Literal     103
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     103
CRD$K_UDA_1_EVDLB  Literal     103
CRD$K_UDA_1_EVDLC  Literal     103
CRD$K_UDA_1_EVDLD  Literal     103
CRD$K_UDA_1_EVRLA_0  Literal     103
CRD$K_UDA_1_EVRLA_1  Literal     103
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     103
CRD$K_UDA_2_EVDLB  Literal     103
CRD$K_UDA_2_EVDLC  Literal     103
CRD$K_UDA_2_EVDLD  Literal     103
CRD$K_UDA_2_EVRLA_0  Literal     103
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     103
CRD$K_UDA_3_EVDLB  Literal     103
CRD$K_UDA_3_EVDLC  Literal     103
CRD$K_UDA_3_EVDLD  Literal     103
CRD$K_UDA_3_EVRLA_0  Literal     103
CRD$K_UDA_3_EVRLA_1  Literal     103
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     103
CRD$PRINT_COMMAND_TABLE  ExtRout  Lib01   157c
CRD$PRINT_HARDWARE_SUPPORT ExtRout  Lib01   169c
CRD$PRINT_PTABLES  ExtRout  Lib01   161c
CRD$SET_CTRL_C_HANDLERS  ExtRout  Lib01   208c
DSA$AL_APTMAIL   Bind       104   104
DSA$GL_APTCOM   Bind       104
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DSA$GL_DEVLEN    Bind         104
DSA$GL_DEVNAM    Bind         104
DSA$GL_ERRNO    Bind         104
DSA$GL_EVENT    Bind         104
DSA$GL_FLAGS    Bind         104   172     173     203     211     212     213     214     216     219     220
              221     222     223     224     225     226     227     228     229     230
              231     232     233     234
DSA$GL_MSGPTR    Bind         104
DSA$GL_MSGTYP    Bind         104
DSA$GL_PASSES    Bind         104
DSA$GL_PASSNO    Bind         104
DSA$GL_SECTNO    Bind         104
DSA$GL_SID    Bind         104
DSA$GL_SUBTNO    Bind         104
DSA$GL_TESTNO    Bind         104
DSA$GL_UNITS    Bind         104
DSA$V_APT    Macro    Lib03   219
DSA$V_BELL    Macro    Lib03   220
DSA$V_BINARY    Macro    Lib03   172     211
DSA$V_CRD_AUTOTEST_OFF  Macro    Lib03   173     203     212
DSA$V_CRD_MENUTEST_OFF  Macro    Lib03   213
DSA$V_CRD_MENUTEST_ON  Macro    Lib03   214
DSA$V_HALT    Macro    Lib03   221
DSA$V_IE1    Macro    Lib03   222
DSA$V_IE2    Macro    Lib03   223
DSA$V_IE3    Macro    Lib03   224
DSA$V_IES    Macro    Lib03   225
DSA$V_LOOP    Macro    Lib03   226
DSA$V_NORPT    Macro    Lib03   227
DSA$V_OPER    Macro    Lib03   228
DSA$V_PROMPT    Macro    Lib03   229
DSA$V_QA    Macro    Lib03   230
DSA$V_QUICK    Macro    Lib03   231
DSA$V_SEARCH    Macro    Lib03   232
DSA$V_TRACE    Macro    Lib03   233
DSA$V_VERIFY    Macro    Lib03   234
DS_FLAGS    Local       201
GET1ST_    Macro    Lib04   103
GET2ND_    Unbound          103
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal    103
HS$K_ACTION_ADVANCE_GENERAL_COM Literal    103
HS$K_ACTION_ADVANCE_STATE Literal    103
HS$K_ACTION_CHANGE_TABLE Literal    103
HS$K_ACTION_DIAGNOSTIC_ERROR Literal    103
HS$K_ACTION_DONE_GENERAL_COMS Literal    103
HS$K_ACTION_DO_NOTHING  Literal    103
HS$K_ACTION_DUMMY_3  Literal    103
HS$K_ACTION_DUMMY_4  Literal    103
HS$K_ACTION_DUMMY_5  Literal    103
HS$K_ACTION_DUMMY_6  Literal    103
HS$K_ACTION_INIT_HS  Literal    103
HS$K_ACTION_INTERNAL_ERROR Literal    103
HS$K_ACTION_LAST_ACTION  Literal    103
```

Symbol	Type	Defined	Referenced ...
HS$K_ACTION_LOAD_ERROR	Literal	103	
HS$K_ACTION_LOAD_GENERAL_COM	Literal	103	
HS$K_ACTION_LOAD_LOAD_COM	Literal	103	
HS$K_ACTION_LOAD_SELECT_COM	Literal	103	
HS$K_ACTION_LOAD_SET_EVENT_COM	Literal	103	
HS$K_ACTION_LOAD_SET_FLAGS_COM	Literal	103	
HS$K_ACTION_LOAD_START_COM	Literal	103	
HS$K_ACTION_PREPARATION_ERROR	Literal	103	
HS$K_ACTION_START_ERROR	Literal	103	
HS$K_STATE_ADVANCE_DIAGNOSTIC	Literal	103	
HS$K_STATE_ADVANCE_GENERAL_COM	Literal	103	
HS$K_STATE_CHANGE_TABLE	Literal	103	
HS$K_STATE_DIAGNOSTIC_ERROR	Literal	103	
HS$K_STATE_DIAGNOSTIC_RUNNING	Literal	103	
HS$K_STATE_DONE_GENERAL_COMS	Literal	103	
HS$K_STATE_DUMMY_1	Literal	103	
HS$K_STATE_DUMMY_2	Literal	103	
HS$K_STATE_DUMMY_3	Literal	103	
HS$K_STATE_DUMMY_4	Literal	103	
HS$K_STATE_DUMMY_6	Literal	103	
HS$K_STATE_INIT_HS	Literal	103	
HS$K_STATE_INTERNAL_ERROR	Literal	103	
HS$K_STATE_LAST_STATE	Literal	103	
HS$K_STATE_LOAD_ERROR	Literal	103	
HS$K_STATE_LOAD_GENERAL_COM	Literal	103	
HS$K_STATE_LOAD_LOAD_COM	Literal	103	
HS$K_STATE_LOAD_SELECT_COM	Literal	103	
HS$K_STATE_LOAD_SET_EVENT_COM	Literal	103	
HS$K_STATE_LOAD_SET_FLAGS_COM	Literal	103	
HS$K_STATE_LOAD_START_COM	Literal	103	
HS$K_STATE_PREPARATION_ERROR	Literal	103	
HS$K_STATE_START_ERROR	Literal	103	
MEN$K_HS_STATE_TABLE	Literal	103	
MEN$K_MM_STATE_TABLE	Literal	103	
MEN$K_TQ_STATE_TABLE	Literal	103	
MENU$K_EXIT_AUTOSIZER_ERROR	Literal	103	
MENU$K_EXIT_INTERNAL_ERROR	Literal	103	
MENU$K_EXIT_LAST_REASON	Literal	103	
MENU$K_EXIT_OPERATOR_ABORT	Literal	103	
MENU$K_EXIT_OPERATOR_STOP	Literal	103	
MENU$K_EXIT_SUP_FILE_LOAD_ERR	Literal	103	
MENU$K_EXIT_SUP_FILE_NOT_FOUND	Literal	103	
MM$K_ACTION_ADVANCE_STATE	Literal	103	
MM$K_ACTION_AUTOSIZER_ERROR	Literal	103	
MM$K_ACTION_BUILD_DDBS	Literal	103	
MM$K_ACTION_BUILD_HSTS	Literal	103	
MM$K_ACTION_CHANGE_TABLE	Literal	103	
MM$K_ACTION_DONE_INITS	Literal	103	
MM$K_ACTION_DO_NOTHING	Literal	103	
MM$K_ACTION_DUMMY_3	Literal	103	
MM$K_ACTION_DUMMY_4	Literal	103	
MM$K_ACTION_DUMMY_5	Literal	103	
MM$K_ACTION_DUMMY_6	Literal	103	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_ACTION_DUMMY_7  Literal    103
MM$K_ACTION_DUMMY_8  Literal    103
MM$K_ACTION_INTERNAL_ERROR Literal    103
MM$K_ACTION_LAST_ACTION  Literal    103
MM$K_ACTION_LOAD_AUTOSIZER Literal    103
MM$K_ACTION_MENU_PROMPT  Literal    103
MM$K_ACTION_PRINT_MENU  Literal    103
MM$K_ACTION_PRINT_MENU_HEADING Literal    103
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal    103
MM$K_ACTION_START_AUTOSIZER Literal    103
MM$K_STATE_AUTOSIZER_ERROR Literal    103
MM$K_STATE_AUTOSIZER_RUNNING Literal    103
MM$K_STATE_BUILD_DDBS  Literal    103
MM$K_STATE_BUILD_HSTS  Literal    103
MM$K_STATE_CHANGE_TABLE  Literal    103
MM$K_STATE_DONE_INITS  Literal    103
MM$K_STATE_DUMMY_1  Literal    103
MM$K_STATE_DUMMY_2  Literal    103
MM$K_STATE_DUMMY_3  Literal    103
MM$K_STATE_DUMMY_5  Literal    103
MM$K_STATE_DUMMY_7  Literal    103
MM$K_STATE_DUMMY_8  Literal    103
MM$K_STATE_INTERNAL_ERROR Literal    103
MM$K_STATE_LAST_STATE  Literal    103
MM$K_STATE_LOAD_AUTOSIZER Literal    103
MM$K_STATE_MENU_PROMPT  Literal    103
MM$K_STATE_PRINT_MENU  Literal    103
MM$K_STATE_PRINT_MENU_HEADING Literal    103
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal    103
MM$K_STATE_START  Literal    103
MM$K_STATE_START_AUTOSIZER Literal    103
MODNAM    Macro    Lib02   120
NUL2ND_     Macro    Lib04   103
OFF    Literal  Lib01   211     212     213     214     219     220     221     222     223     224
             225     226     227     229     230     231     232     233     234
ON    Literal  Lib01   172     173     228
TQ$K_ACTION_ADVANCE_STATE Literal    103
TQ$K_ACTION_CHANGE_TABLE Literal    103
TQ$K_ACTION_DONE_TEST_QUEUE Literal    103
TQ$K_ACTION_DO_NOTHING  Literal    103
TQ$K_ACTION_DUMMY_3  Literal    103
TQ$K_ACTION_DUMMY_4  Literal    103
TQ$K_ACTION_DUMMY_5  Literal    103
TQ$K_ACTION_GET_DDB  Literal    103
TQ$K_ACTION_INIT_TQ  Literal    103
TQ$K_ACTION_INTERNAL_ERROR Literal    103
TQ$K_ACTION_LAST_ACTION  Literal    103
TQ$K_STATE_CHANGE_TABLE  Literal    103
TQ$K_STATE_DONE_TEST_QUEUE Literal    103
TQ$K_STATE_DUMMY_1  Literal    103
TQ$K_STATE_DUMMY_2  Literal    103
TQ$K_STATE_DUMMY_3  Literal    103
TQ$K_STATE_DUMMY_4  Literal    103
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
TQ$K_STATE_DUMMY_5  Literal    103
TQ$K_STATE_GET_DDB  Literal    103
TQ$K_STATE_INIT_TQ  Literal    103
TQ$K_STATE_INTERNAL_ERROR Literal    103
TQ$K_STATE_LAST_STATE  Literal    103



  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDINIT.B32;1
    2  Module CRDINIT
   83  Library #1 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   86  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   89  Library #3 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   92  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  239  Eludom CRDINIT


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     a Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare dec aration


;    Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486         22          4          62          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923         21          2          94          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2       18619          3           0       1000          00:04.4
```

```
;     COMMAND QUALIFIERS

; BLISS CRDINIT.B32/LIST=CRDINIT.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDINIT.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  197 code + 16 data bytes
; Run Time:    00:20.0
; Elapsed Time:    00:37.8
; Lines/CPU Min:     718
; Lexemes/CPU-Min: 53735
; Memory Used: 167 pages
; Compilation Complete
```

```
;    0001   0 %TITLE '*** CRDINTRFC Module'
;    0002   0 MODULE CRDINTRFC ( IDENT='07', ZIP, OPTLEVEL=3) =
;    0003   0
;    0004   0 !++
;    0005   0 ! COPYRIGHT (c) 1980, 1981, 1985
;    0006   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0007   0 !
;    0008   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0009   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0010   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0011   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0012   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0013   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0014   0 ! REMAIN IN DEC.
;    0015   0 !
;    0016   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0017   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0018   0 ! CORPORATION.
;    0019   0 !
;    0020   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0021   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0022   0 !-
;    0023   0
;    0024   0 !++
;    0025   0 !   History:
;    0026   0 !
;    0027   0 ! 00 Jack Stansbury, 29-March-1982
;    0028   0 !  Created module
;    0029   0 !
;    0030   0 ! 01 Jack Stansbury, 15-July-1982, Version 1.0
;    0031   0 !  Added code to support CRD Menu Test.
;    0032   0 !
;    0033   0 ! 02 Jack Stansbury, August 31, 1982, Version 1.1
;    0034   0 !  Added code to allow Menu Test to get User_Prompts without specifying them in the
;    0035   0 !  State Tables. This is primarily for the ^C User_Prompt, which can happen at any time.
;    0036   0 !
;    0037   0 ! 03 John Ciukaj 5-Apr-1983 Version 1.2
;    0038   0 !  - Changed references to CRD bits in DSA Flags to
;    0039   0 !    distinguish between Online and Offline.
;    0040   0 !
;    0041   0 ! 04 Scott Cohen 16-June-1983 Version 1.2
;    0042   0 !  Added code to allow Auto Test to get User_Prompts without specifying them in the
;    0043   0 !  State Tables. This is primarily for the soft console support.
;    0044   0 !
;    0045   0 ! 05 Ronald W. Parker 27-SEP-1984
;    0046   0 !  Cleaned up, added CRD MENU ONLINE mechanism, defined
;    0047   0 !  the undefined return value when cases fail.
;    0048   0 !
;    0049   0 ! 06 Ronald W. Parker 5-DEC-1984
;    0050   0 !  Added CRD$LOG_ routines
;    0051   0 !
;    0052   0 ! 07 Ronald W. Parker 14-March-1984
;    0053   0 !  - Add 810 action on offline and check for 730 offline
;    0054   0 !--
```

```
;    0055  0
;    0056  0 %SBTTL 'Libraries'
;    0057  1 BEGIN     ! Begin the CRDINTRFC module
;    0058  1
;    0059  1 LIBRARY '$CRD';
;    0060  1 LIBRARY '$DS';
;    0061  1 LIBRARY '$Diag';
;    0062  1 LIBRARY 'SYS$Library:LIB';
;    0063  1
;    0064  1
;    0065  1 %SBTTL 'Table of Contents'
;    0066  1
;    0067  1 FORWARD ROUTINE
;    0068  1  ! The interface routine between the Resident-DS and CRD-Loadable
;    0069  1  CRD$DS_Interface : Addressing_Mode (Long_Relative),
;    0070  1
;    0071  1  ! CRD logging routines
;    0072  1  CRD$Log_Start  : ADDRESSING_MODE (LONG_RELATIVE),      !006
;    0073  1  CRD$Log_Stop   : ADDRESSING_MODE (LONG_RELATIVE),      !006
;    0074  1  CRD$Log_Write  : ADDRESSING_MODE (LONG_RELATIVE);      !006
;    0075  1
;    0076  1 %SBTTL 'Included Macros and Literals'
;    0077  1
;    0078  1  $DS_DSADef;   ! Define the DSA bits       ![01]
;    0079  1  $CRD_Literals;   ! Define some literals for CRD
;    0080  1  $DS_TypeDef;   ! Define the typecodes       ![02]
;    0081  1
;    0082  1
;    0083  1 %SBTTL 'Psect Definitions'
;    0084  1
;    0085  1 PSECT
;    0086  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0087  1
;    0088  1 PSECT
;    0089  1     CODE    - Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0090  1
;    0091  1 PSECT
;    0092  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0093  1
;    0094  1 PSECT
;    0095  1     GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0096  1
;    0097  1
;    0098  1 %SBTTL 'Module-Global Static Storage'
;    0099  1
;    0100  1 ModNam ('CRDINTRFC');  ! Module name for ErrSup macro
;    0101  1
; M  0102  1 KEYWORDMACRO $LOAD_ASCID_S ( NAM,STR ) =       ![06]
; M  0103  1     Nam[0,00,16,0] = %CHARCOUNT(STR);        ![06]
; M  0104  1     Nam[0,16,08,0] = DSC$K_DTYPE_T;
; M  0105  1     Nam[0,24,08,0] = DSC$K_CLASS_S;
;    0106  1     Nam[1,00,32,0] = UPLIT (%ASCII STR)%;         ![06]
;    0107  1
;    0108  1 OWN                 ![06]
;    0109  1     LOG_FAB : $FAB ( RFM=VAR, RAT=CR ),          ![06]
```

```
;    0110  1     LOG_RAB : $RAB ();              ![06]
;    0111  1
;    0112  1 GLOBAL                  ![02]
;    0113  1  CRD$GB_User_Prompt_Okay : BYTE INITIAL (False);        ![02]
;    0114  1       ! If set to True, then User_Prompt typecodes will not result in a call ![02]
;    0115  1       ! to the MEN$Turing_Machine routine.     ![02]
```

```
;    0116  1 %SBTTL 'CRD$DS_Interface Routine'
;    0117  1
;    0118  1 GLOBAL ROUTINE CRD$DS_Interface (Function, Code, Length, Address) =
;    0119  1
;    0120  1 !++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;    0121  1 !++
;    0122  1 ! Functional Description:
;    0123  1 !
;    0124  1 !   This routine is the interface routine between the Resident-Diagnostic
;    0125  1 !   Supervisor and th Loadable-Customer Runnable Diagnostic package. This
;    0126  1 !   routine is the one AND ONLY routine that the Resident-DS routines call when
;    0127  1 !   interfacing with the CRD package is required. This routine will dispatch to
;    0128  1 !   several different CRD routines, based on the value of the first parameter.
;    0129  1 !
;    0130  1 ! Formal Parameters:
;    0131  1 !
;    0132  1 ! Function: Determines what type of CRD call this was. It could be
;    0133  1 !   a call to the CRD package because of a DS Print
;    0134  1 !   statement, or because CRD should initialize itself.
;    0135  1 ! Code:  Either a TypeCode, as defined by the $DS_TypeDef macro,
;    0136  1 !   or a particular CRD function code, such as
;    0137  1 !   initialization, etc.
;    0138  1 ! Length:  The length of the string that is about to be printed.
;    0139  1 !   This could also be CLI buffer length.
;    0140  1 ! Address: The address of the string that is about to be printed.
;    0141  1 !   This could also be the CLI buffer address.
;    0142  1 !
;    0143  1 ! Completion Codes:
;    0144  1 !
;    0145  1 ! 0 => Print what was about to be printed by the Resident-DS.
;    0146  1 ! 1 => Branch around the actual outputting of the string.
;    0147  1 ! 2 => Invalid FUNCTION parameter passed to CRDINTRFC
;    0148  1 ! 4 => Illegal mode found in DSA_Extract
;    0149  1 ! 6 => Invalid CODE parameter passed to CRDINTRFC
;    0150  1 ! 8 => Illegal mode found in DSA_Extract
;    0151  1 ! 12=> Autotest offline mode but not 730 or 810 processor
;    0152  1 !--
;    0153  1 !----------------------------------------------------------------------------
;    0154  1
;    0155  2 BEGIN
;    0156  2
;    0157  2 !++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;    0158  2 !+
;    0159  2 !   Variable definitions
;    0160  2 !-
;    0161  2 !----------------------------------------------------------------------------
;    0162  2
;    0163  2 REGISTER
;    0164  2     DSA_Extract, P1, P2, Return_Value;    ! [07]
;    0165  2
;    0166  2 OWN
;    0167  2     Command : BLOCK [2,LONG];
```

```
;    0168  2
;    0169  2 !**************************************************************************
;    0170  2 !+
;    0171  2 !   Store the mode control bits in a contiguous bit stream for easier
;    0172  2 !   reference later on. DSA_Extract, bits 0-2, will contain the DSA
;    0173  2 !   bits. Bits 3-7 of DSA_Extract will be zero.
;    0174  2 !
;    0175  2 !    Bit 0 will be Autotest offline
;    0176  2 !    Bit 1 will be Menutest offline
;    0177  2 !    Bit 2 will be Menutest online
;    0178  2 !-
;    0179  2 !**************************************************************************
;    0180  2
;    0181  2 DSA_Extract = 0;      ! [05]
;    0182  2 DSA_Extract<0,1,0> = .DSA$V_CRD_Autotest_off;   ! [05]
;    0183  2 DSA_Extract<1,1,0> = .DSA$V_CRD_Menutest_off;   ! [05]
;    0184  2 DSA_Extract<2,1,0> = .DSA$V_CRD_Menutest_On;   ! [05]
;    0185  2
;    0186  2 !**************************************************************************
;    0187  2 !+
;    0188  2 !   Now decide the action to take based on the function passed.
;    0189  2 !-
;    0190  2 !**************************************************************************
;    0191  2
;    0192  2 CASE .Function FROM 0 TO (CRD$K_Last_Function - 1) OF
;    0193  2     SET
;    0194  2     [CRD$K_Hook_Point]:
;    0195  3  BEGIN
;    0196  3  CASE .Code FROM 0 TO (CRD$K_Last_Hook_Point - 1) OF
;    0197  3      SET
;    0198  3      [CRD$K_CRD_Initialization]:
;    0199  4   BEGIN
;    0200  4
;    0201  4 !    .    .    .   Initialize CRD according to the mode
;    0202  4
;    0203  4       CASE .DSA_Extract FROM 1 TO 4 OF   ! [05]
;    0204  4       SET        ! [05]
;    0205  4       [ 1    ] :
;    0206  4 !    .    .    .    .    .   Autotest offline mode
;    0207  4 !    .    .    .    .    .   IF 11/730 processor, do autotest offline init
;    0208  5    IF (.DSA$GL_SID<24,8,0> EQL PR$_SID_TYP730)
;    0209  4        THEN Return_Value = CRD$CRD_Initialization()! [07]
;    0210  4        ELSE Return_Value = 1;   ! [07]
;    0211  4       [ 2,4 ] :
;    0212  4 !    .    .    .    .    .   Menutest offline and mode
;    0213  4    Return_Value=MEN$Menu_Fnctst_Init(.DSA_Extract);! [05]
;    0214  4       [ INRANGE  ] : Return_Value = 4; ! ERROR  ! [05]
;    0215  4       [ OUTRANGE ] : Return_Value = 4; ! ERROR  ! [05]
;    0216  4          TES;      ! [05]
;    0217  3   END;       ! [05]
;    0218  3          [ OUTRANGE ] : Return_Value = 6; ! ERROR   ! [05]
;    0219  3      TES;
;    0220  2  END;
```

```
;    0221  2     [CRD$K_TypeCode]:
;    0222  2
;    0223  2 !    .   IF its okay to prompt and prompt requested
;    0224  2 !    .   .   THEN return 0 so VDS behaves as if CRD doens't exsist
;    0225  2 !    .   .   ELSE Setup parameters and execute CRD code.
;    0226  2
;    0227  3  IF ((.Code EQL DS$K_Type_User_Prompt) AND (.CRD$GB_User_Prompt_Okay))
;    0228  2  THEN
;    0229  2      Return_Value = 0
;    0230  2      ELSE
;    0231  3      BEGIN
;    0232  3
;    0233  3 !    .   .   IF executing via a command file
;    0234  3 !    .   .   .   THEN fudge the prompt
;    0235  3 !    .   .   .   ELSE pass normal values
;    0236  3
;    0237  4      IF (.Code EQL DS$K_Type_Script_Echo)
;    0238  3      THEN
;    0239  4          BEGIN
;    0240  4          P1 = DS$K_Type_DS_Prompt;    ! [05]
;    0241  4          P2 = 80;      ! [05]
;    0242  4          END
;    0243  3      ELSE
;    0244  4   BEGIN
;    0245  4   P1 = .Code;      ! [05]
;    0246  4   P2 = .Length;      ! [05]
;    0247  3   END;
;    0248  3
```

```
;    0249  3
;    0250  3 !    .    .    Now select the appropriate turing machine and execute
;    0251  3
;    0252  3       CASE .DSA_Extract FROM 1 TO 4 OF     ! [05]
;    0253  3           SET          ! [05]
;    0254  3           [ 1    ] :       ! [05]
;    0255  4         BEGIN
;    0256  4 !    .    .    .    .    Autotest offline
;    0257  4 !    .    .    .    .    IF 11/730, do autotest offline stuff
;    0258  4         IF (.DSA$GL_SID<24,8,0> EQL PR$_SID_TYP730) THEN ! [07]
;    0259  4             Return_Value=CRD$Turing_Machine (.P1,.P2,.Address);! [05]
;    0260  4
;    0261  4 !    .    .    .    .    IF CPU=820 and start typecode passed THEN do nothing
;    0262  4         IF  (.DSA$GL_SID<24,8,0> EQL PR$_SID_TYP8SS) AND
;    0263  5      ( .CODE EQL DS$K_TYPE_DS_START)
;    0264  4         THEN Return_Value=CRD$K_Success;    ! [07]
;    0265  4
;    0266  4 !    .    .    .    .    IF CPU=820/8nn and Prompt typecode passed THEN aVDS_SCRIPT
;    0267  4 !    .    .    .    .    Special function to execute a command file on 820 CPUs
;    0268  4 !    .    .    .    .    Also clear any CRD bits, so as not to come back to CRD
;    0269  5         IF  ((.DSA$GL_SID<24,8,0> EQL PR$_SID_TYP8SS)  OR
;    0270  4       (.DSA$GL_SID<24,8,0> EQL PR$_SID_TYP8NN)) AND
;    0271  5       ( .CODE EQL DS$K_TYPE_DS_PROMPT)
;    0272  4         THEN
;    0273  5      BEGIN
;    0274  5      $LOAD_ASCID_S(NAM=Command, STR='aVDS_SCRIPT.COM');
;    0275  5      CH$COPY (.Command[0,0,16,0],
;    0276  5          CH$PTR(.Command[1,0,32,0]),
;    0277  5          %C' ',
;    0278  5          .P2,
;    0279  5          CH$PTR(.Address)
;    0280  5          );
;    0281  5      DSA$V_CRD_Autotest_Off = 0;
;    0282  5      DSA$V_CRD_Menutest_Off = 0;
;    0283  5         DSA$V_CRD_Menutest_On  = 0;
;    0284  5      DSA$V_CRD_Trace = 0;
;    0285  5      $CRD_Return_Length_status (CRD$K_Success);
;    0286  4      END;
;    0287  4         Return_Value = 12;      ! [07]
;    0288  3         END;
;    0289  3                    [ 2,4 ] :       ! [05]
;    0290  3 !    .    .    .    .    Menutest offline and online
;    0291  3         Return_Value=MEN$Turing_Machine (.P1,.P2,.Address); ! [05]
;    0292  3            [ INRANGE  ] : Return_Value = 8; ! ERROR  ! [05]
;    0293  3            [ OUTRANGE ] : Return_Value = 8; ! ERROR  ! [05]
;    0294  3            TES;         ! [05]
;    0295  2       END;
;    0296  2      [ OUTRANGE ] : Return_Value=2; ! ERROR
;    0297  2      TES;
;    0298  2 RETURN (.Return_Value);
;    0299  1 END;      ! Routine CRD$DS_Interface


            .TITLE  CRDINTRFC *** CRDINTRFC Module
            .IDENT  \07\
```

```
                .PSECT  WORK,NOEXE,2

        03  00000 LOG_FAB:.BYTE   3                ;
        50  00001     .BYTE   80                 ;
             0000  00002     .WORD   0                  ;
         00000000  00004     .LONG   0                  ;
         00000000  00008     .LONG   0                  ;
         00000000  0000C     .LONG   0                  ;
         00000000  00010     .LONG   0                  ;
             0000  00014     .WORD   0                  ;
        02  00016     .BYTE   2                  ;
        00  00017     .BYTE   0                  ;
         00000000  00018     .LONG   0                  ;
        00  0001C     .BYTE   0                  ;
        00  0001D     .BYTE   0                  ;
        02  0001E     .BYTE   2                  ;
        02  0001F     .BYTE   2                  ;
         00000000  00020     .LONG   0                  ;
         00000000  00024     .LONG   0                  ;
         00000000  00028     .LONG   0                  ;
         00000000  0002C     .LONG   0                  ;
         00000000  00030     .LONG   0                  ;
        00  00034     .BYTE   0                  ;
        00  00035     .BYTE   0                  ;
             0000  00036     .WORD   0                  ;
         00000000  00038     .LONG   0                  ;
             0000  0003C     .WORD   0                  ;
        00  0003E     .BYTE   0                  ;
        00  0003F     .BYTE   0                  ;
         00000000  00040     .LONG   0                  ;
         00000000  00044     .LONG   0                  ;
             0000  00048     .WORD   0                  ;
        00  0004A     .BYTE   0                  ;
        00  0004B     .BYTE   0                  ;
         00000000  0004C     .LONG   0                  ;
        01  00050 LOG_RAB:.BYTE   1                ;
        44  00051     .BYTE   68                 ;
             0000  00052     .WORD   0                  ;
         00000000  00054     .LONG   0                  ;
         00000000  00058     .LONG   0                  ;
         00000000  0005C     .LONG   0                  ;
             0000# 00060     .WORD   0[3]                ;
             0000  00066     .WORD   0                  ;
         00000000  00068     .LONG   0                  ;
             0000  0006C     .WORD   0                  ;
        00  0006E     .BYTE   0                  ;
        00  0006F     .BYTE   0                  ;
             0000  00070     .WORD   0                  ;
             0000  00072     .WORD   0                  ;
         00000000  00074     .LONG   0                  ;
         00000000  00078     .LONG   0                  ;
         00000000  0007C     .LONG   0                  ;
         00000000  00080     .LONG   0                  ;
        00  00084     .BYTE   0                  ;
```

```
        00  00085     .BYTE   0           ;
        00  00086     .BYTE   0           ;
        00  00087     .BYTE   0           ;
         00000000  00088     .LONG   0           ;
         00000000  0008C     .LONG   0           ;
         00000000  00090     .LONG   0           ;
        00  00094 CRD$GB_USER_PROMPT_OKAY::
            .BYTE   0           ;
            00095   .BLKB   3
            00098 COMMAND:.BLKB   8

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

       43  46  52  54  4E  49  44  52  43  09  00000 P.AAA:  .ASCII  <9>\CRDINTRFC\          ;
            0000A   .BLKB   2
4D  4F  43  2E  54  50  49  52  43  53  5F  53  44  56  40  0000C P.AAB:  .ASCII  \aVDS_SCRIPT.COM\<0>          ;
        00  0001B               ;

           DSA$AL_APTMAIL=     65024
           DSA$GL_FLAGS=       65024
           DSA$GL_APTCOM=      65028
           DSA$GL_PASSES=      65032
           DSA$GL_UNITS=       65036
           DSA$GL_SECTNO=      65040
           DSA$GL_SID=        65044
           DSA$GL_MSGTYP=      65088
           DSA$GL_ERRNO=       65092
           DSA$GL_EVENT=       65096
           DSA$GL_SUBTNO=      65100
           DSA$GL_TESTNO=      65104
           DSA$GL_PASSNO=      65108
           DSA$GL_DEVLEN=      65112
           DSA$GL_DEVNAM=      65116
           DSA$GL_MSGPTR=      65128
           $MODULE=        P.AAA
            .EXTRN  CRD$DS_INTERFACE
            .EXTRN  CRD$CRD_INITIALIZATION
            .EXTRN  MEN$MENU_FNCTST_INIT
            .EXTRN  CRD$TURING_MACHINE
            .EXTRN  MEN$TURING_MACHINE

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              007C 00000     .ENTRY  CRD$DS_INTERFACE, Save R2,R3,R4,R5,R6       ; 0118
             52  D4 00002     CLRL    DSA_EXTRACT             ; 0181
50 0000FE01   9F         01      03  EF 00004    EXTZV   #3, #1, a#^X0000FE01, R0        ; 0182
52            01        00      50  F0 0000D    INSV    R0, #0, #1, DSA_EXTRACT         ;
52            01        01 0000FE02   9F  F0 00012    INSV    a#^X0000FE02, #1, #1, DSA_EXTRACT       ; 0183
50 0000FE02   9F         01      02  EF 0001B    EXTZV   #2, #1, a#^X0000FE02, R0        ; 0184
52            01        02      50  F0 00024    INSV    R0, #2, #1, DSA_EXTRACT         ;
            01         00 04    AC  CF 00029    CASEL   FUNCTION, #0, #1            ; 0192
          0050    000A     0002E 1$:   .WORD   2$-1$,-             ;
            10$-1$              ;
            56      02  D0 00032    MOVL    #2, RETURN_VALUE            ; 0296
          0107  31 00035    BRW     23$             ;
```

```
          00        00 08   AC  CF 00038 2$:   CASEL   CODE, #0, #0          ; 0196
        0008      0003D 3$:    .WORD   4$-3$                ;
          56        06  D0 0003F    MOVL    #6, RETURN_VALUE          ; 0218
        00FA  31 00042    BRW     23$                ;
          03        01      52  CF 00045 4$:   CASEL   DSA_EXTRACT, #1, #3          ; 0203
  0023        002F      0023    000A     00049 5$:   .WORD   6$-5$,-          ;
           8$-5$,-          ;
           9$-5$,-          ;
           8$-5$           ;
          25  11 00051    BRB     9$              ; 0215
          03 0000FE17   9F  91 00053 6$:   CMPB    @#^X0000FE17, #3          ; 0208
          0A  12 0005A    BNEQ    7$              ;
      00000000G  EF       00  FB 0005C    CALLS   #0, CRD$CRD_INITIALIZATION        ; 0209
          00D1  31 00063    BRW     21$                ;
          56        01  D0 00066 7$:   MOVL    #1, RETURN_VALUE          ; 0210
          00D3  31 00069    BRW     23$              ; 0208
          52  DD 0006C 8$:   PUSHL   DSA_EXTRACT            ; 0213
      00000000G  EF       01  FB 0006E    CALLS   #1, MEN$MENU_FNCTST_INIT        ;
          00BF  31 00075    BRW     21$                ;
          56        04  D0 00078 9$:   MOVL    #4, RETURN_VALUE          ; 0214
          00C1  31 0007B    BRW     23$              ; 0196
          54 08   AC  D0 0007E 10$:   MOVL    CODE, R4            ; 0227
          02        54  D1 00082    CMPL    R4, #2            ;
          0C  12 00085    BNEQ    11$                ;
          05 00000000'  EF  E9 00087    BLBC    CRD$GB_USER_PROMPT_OKAY, 11$        ;
          56  D4 0008E    CLRL    RETURN_VALUE            ; 0229
          00AC  31 00090    BRW     23$                ;
          21        54  D1 00093 11$:   CMPL    R4, #33           ; 0237
          09  12 00096    BNEQ    12$                ;
          50        01  D0 00098    MOVL    #1, P1            ; 0240
          53 50   8F  9A 0009B    MOVZBL  #80, P2            ; 0241
          07  11 0009F    BRB     13$              ; 0237
          50        54  D0 000A1 12$:   MOVL    R4, P1            ; 0245
          53 0C   AC  D0 000A4    MOVL    LENGTH, P2            ; 0246
          03          01      52  CF 000A8 13$:   CASEL   DSA_EXTRACT, #1, #3          ; 0252
  007F        0090        007F    000B     000AC 14$:   .WORD   15$-14$,-          ;
           20$-14$,-          ;
           22$-14$,-          ;
           20$-14$           ;
         0085  31 000B4    BRW     22$              ; 0293
          52 0000FE17   9F  9A 000B7 15$:   MOVZBL  @#^X0000FE17, R2          ; 0258
          03        52  91 000BE    CMPB    R2, #3            ;
          0F  12 000C1    BNEQ    16$              ;
      10  AC  DD 000C3    PUSHL   ADDRESS             ; 0259
          09  BB 000C6    PUSHR   #^M<R0,R3>          ;
       00000000G  EF       03  FB 000C8    CALLS   #3, CRD$TURING_MACHINE        ;
          56        50  D0 000CF    MOVL    R0, RETURN_VALUE          ;
          50  D4 000D2 16$:   CLRL    R0              ; 0262
          05        52  91 000D4    CMPB    R2, #5            ;
          0A  12 000D7    BNEQ    17$              ;
          50  D6 000D9    INCL    R0              ;
          1D        54  D1 000DB    CMPL    R4, #29           ; 0263
          03  12 000DE    BNEQ    17$              ;
          56        01  D0 000E0    MOVL    #1, RETURN_VALUE          ; 0264
          05        50  E8 000E3 17$:   BLBS    R0, 18$           ; 0269
```

```
                06      52  91 000E6     CMPB    R2, #6          ; 0270
                3B  12 000E9     BNEQ    19$             ;
                01      54  D1 000EB 18$:   CMPL    R4, #1          ; 0271
                36  12 000EE     BNEQ    19$             ;
            00000000'  EF 010E000F   8F  D0 000F0    MOVL    #17694735, COMMAND          ; 0274
            00000000'  EF 00000000'  EF  9E 000FB    MOVAB   P.AAB, COMMAND+4        ;
        53          20 00000000'  FF 00000000'  EF  2C 00106    MOVC5   COMMAND, aCOMMAND+4, #32, P2, aADDRESS      ; 0279
          10   BC     00113                   ;
           0000FE01   9F     0708   8F  AA 00115    BICW2   #1800, a#^X0000FE01         ; 0284
                50 00500001   8F  D0 0011E    MOVL    #5242881, R0            ; 0285
             04 00125    RET                 ;
                56      0C  D0 00126 19$:   MOVL    #12, RETURN_VALUE           ; 0287
                14  11 00129     BRB     23$             ; 0252
          10    AC  DD 0012B 20$:   PUSHL   ADDRESS             ; 0291
                09  BB 0012E     PUSHR   #^M<R0,R3>          ;
            00000000G  EF       03  FB 00130    CALLS   #3, MEN$TURING_MACHINE      ;
                56      50  D0 00137 21$:   MOVL    R0, RETURN_VALUE            ;
                03  11 0013A     BRB     23$             ;
                56      08  D0 0013C 22$:   MOVL    #8, RETURN_VALUE            ; 0292
                50      56  D0 0013F 23$:   MOVL    RETURN_VALUE, R0            ; 0298
             04 00142    RET                 ; 0299

; Routine Size:  323 bytes,    Routine Base:  CODE + 0000
```

```
;    0300  1 %TITLE 'CRD$INTRFC module'
;    0301  1 %SBTTL 'CRD$Error_Log routine'
;    0302  1
;    0303  1 GLOBAL ROUTINE CRD$Log_Start =
;    0304  1
;    0305  1 !++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;    0306  1 !+
;    0307  1 !   Function:
;    0308  1 !    This routine will delete the old SYS$MAINTENANCE:CRD.LOG files
;    0309  1 !    and then create a new SYS$MAINTENANCE:CRD.LOG file with header
;    0310  1 !
;    0311  1 !   Calling sequence:
;    0312  1 !    CALLS #0,L^CRD$Log_Start
;    0313  1 !
;    0314  1 !   Return Status:
;    0315  1 !    All operation, pass or fail return success
;    0316  1 !-
;    0317  1 !----------------------------------------------------------------------
;    0318  1
;    0319  2 BEGIN
;    0320  2 OWN                 ! [06]
;    0321  2     Desc : BLOCK [2,LONG],          ! [06]
;    0322  2     Header_Size : VECTOR [1,WORD],          ! [06]
;    0323  2     Header_Buff : VECTOR [132,BYTE];          ! [06]
;    0324  2
;    0325  2 ! Fill in FAB name
;    0326  2
;    0327  2 Log_RAB[RAB$L_FAB] = Log_FAB; ! This is for Position Independent code     ! [06]
;    0328  2
;    0329  2 ! Delete any old copies of the file
;    0330  2
;    0331  2 Log_FAB[FAB$L_FNA] = UPLIT(%ASCII 'SYS$MAINTENANCE:CRD.LOG;*' );     ! [06]
;    0332  2 Log_FAB[FAB$B_FNS] = %CHARCOUNT(  'SYS$MAINTENANCE:CRD.LOG;*' );     ! [06]
;    0333  2 $ERASE (FAB=Log_FAB);            ! [06]
;    0334  2
;    0335  2 ! Create a new blank file
;    0336  2
;    0337  2 Log_FAB[FAB$L_FNA] = UPLIT(%ASCII 'SYS$MAINTENANCE:CRD.LOG' );      ! [06]
;    0338  2 Log_FAB[FAB$B_FNS] = %CHARCOUNT(  'SYS$MAINTENANCE:CRD.LOG' );      ! [06]
;    0339  2 $OPEN  (FAB=Log_FAB);            ! [06]
;    0340  2
;    0341  2 ! Construct the log file page header
;    0342  2
; P  0343  2 $LOAD_ASCID_S (              ! [06]
; P  0344  2     NAM = DESC,              ! [06]
; P  0345  2     STR = '!/^!/ Customer Runnable Diagnostics Log file!10 !%D!/!/'     ! [06]
;    0346  2     );                ! [06]
; P  0347  2 $FAO (                 ! [06]
; P  0348  2     DESC,  ! PIC ASCID string        ! [06]
; P  0349  2     Header_Size, ! Store size of constructed string here       ! [06]
;    0350  2     Header_Buff ); ! Store constructed string here       ! [06]
;    0351  2
;    0352  2 !Write the header record into the log file
;    0353  2
;    0354  2 Log_RAB[RAB$L_RBF] = .Header_Buff;          ! [06]
```

```
;    0355  2 Log_RAB[RAB$W_RSZ] = .Header_Size;          ! [06]
;    0356  2 $WRITE (RAB=Log_RAB);                      ! [06]
;    0357  2 RETURN CRD$K_Success;                      ! [06]
;    0358  1 END;                                       ! [06]



                .PSECT  WORK,NOEXE,2

                000A0 DESC:     .BLKB   8
                000A8 HEADER_SIZE:
                .BLKB   2
                000AA   .BLKB   2
                000AC HEADER_BUFF:
                .BLKB   132

                .PSECT  DATA,NOWRT,NOEXE,  SHR,2

45  43  4E  41  4E  45  54  4E  49  41  4D  24  53  59  53  0001C P.AAC:   .ASCII  \SYS$MAINTENANCE:CRD.LOG;*\<0><0><0>       ;
 00  00  00  2A  3B  47  4F  4C  2E  44  52  43  3A  0002B                 ;
45  43  4E  41  4E  45  54  4E  49  41  4D  24  53  59  53  00038 P.AAD:   .ASCII  \SYS$MAINTENANCE:CRD.LOG\<0>        ;
   00  47  4F  4C  2E  44  52  43  3A  00047                   ;
20  72  65  6D  6F  74  73  75  43  20  2F  21  5E  2F  21  00050 P.AAE:   .ASCII  \!/^!/ Customer Runnable Diagnostics Log \  ;
6F  6E  67  61  69  44  20  65  6C  62  61  6E  6E  75  52  0005F                     ;
      20  67  6F  4C  20  73  63  69  74  73  0006E                       ;
2F  21  2F  21  44  25  21  20  30  31  21  65  6C  69  66  00078     .ASCII  \file!10 !%D!/!/\<0>          ;
       00  00087                 ;

                .EXTRN  SYS$ERASE, SYS$OPEN
                .EXTRN  SYS$FAO, SYS$WRITE

                .PSECT  CODE,NOWRT,  SHR,  PIC,2

      0000 00000     .ENTRY  CRD$LOG_START, Save nothing          ; 0303
00000000'  EF 00000000'  EF  9E 00002     MOVAB   LOG_FAB, LOG_RAB+60          ; 0327
00000000'  EF 00000000'  EF  9E 0000D     MOVAB   P.AAC, LOG_FAB+44          ; 0331
00000000'  EF       19  90 00018     MOVB    #25, LOG_FAB+52          ; 0332
00000000'  EF  9F 0001F     PUSHAB  LOG_FAB                  ; 0333
00000000G  00       01  FB 00025     CALLS   #1, SYS$ERASE                ;
00000000'  EF 00000000'  EF  9E 0002C     MOVAB   P.AAD, LOG_FAB+44          ; 0337
00000000'  EF       17  90 00037     MOVB    #23, LOG_FAB+52          ; 0338
00000000'  EF  9F 0003E     PUSHAB  LOG_FAB                  ; 0339
00000000G  00       01  FB 00044     CALLS   #1, SYS$OPEN                 ;
00000000'  EF 010E0037   8F  D0 0004B     MOVL    #17694775, DESC          ; 0346
00000000'  EF 00000000'  EF  9E 00056     MOVAB   P.AAE, DESC+4                ;
00000000'  EF  9F 00061     PUSHAB  HEADER_BUFF              ; 0350
00000000'  EF  9F 00067     PUSHAB  HEADER_SIZE              ;
00000000'  EF  9F 0006D     PUSHAB  DESC                 ;
00000000G  00       03  FB 00073     CALLS   #3, SYS$FAO                  ;
00000000'  EF 00000000'  EF  D0 0007A     MOVL    HEADER_BUFF, LOG_RAB+40          ; 0354
00000000'  EF 00000000'  EF  B0 00085     MOVW    HEADER_SIZE, LOG_RAB+34          ; 0355
00000000'  EF  9F 00090     PUSHAB  LOG_RAB                  ; 0356
00000000G  00       01  FB 00096     CALLS   #1, SYS$WRITE                ;
    50       01  D0 0009D     MOVL    #1, R0                   ; 0357
 04 000A0     RET                 ; 0358
```

```
; Routine Size:  161 bytes,    Routine Base:  CODE + 0143

;   0359  1
```

```
;    0360  1 GLOBAL ROUTINE CRD$Log_Stop =
;    0361  1
;    0362  1 !**************************************************************************
;    0363  1 !+
;    0364  1 !   Function:
;    0365  1 ! This routine will simply close the SYS$MAINTENACNE:CRD.LOG file
;    0366  1 !
;    0367  1 !   Calling sequence:
;    0368  1 !    CALLS #0,L^CRD$Log_Stop
;    0369  1 !
;    0370  1 !   Return Status:
;    0371  1 !    All operation, pass or fail return success
;    0372  1 !-
;    0373  1 !**************************************************************************
;    0374  1
;    0375  2 BEGIN
;    0376  2
;    0377  2 Log_RAB[RAB$L_FAB] = Log_FAB;          ! [06]
;    0378  2 $CLOSE (FAB=Log_FAB);                 ! [06]
;    0379  2
;    0380  2 RETURN CRD$K_Success;                 ! [06]
;    0381  1 END;                  ! [06]



          .EXTRN  SYS$CLOSE

                0000 00000     .ENTRY  CRD$LOG_STOP, Save nothing        ; 0360
    00000000'  EF 00000000'  EF  9E 00002     MOVAB   LOG_FAB, LOG_RAB+60          ; 0377
    00000000'  EF  9F 0000D     PUSHAB  LOG_FAB                ; 0378
    00000000G  00       01  FB 00013     CALLS   #1, SYS$CLOSE           ;
         50       01  D0 0001A     MOVL    #1, R0                ; 0380
          04 0001D     RET                 ; 0381

; Routine Size:  30 bytes,    Routine Base:  CODE + 01E4


;    0382  1
```

```
;    0383  1
;    0384  1 GLOBAL ROUTINE CRD$Log_Write =
;    0385  1
;    0386  1 !**************************************************************************
;    0387  1 !+
;    0388  1 !   Function:
;    0389  1 ! This routine writes the string passed via Adr and Len into
;    0390  1 ! SYS$MAINTENACNE:CRD.LOG.
;    0391  1 !
;    0392  1 !   Calling sequence:
;    0393  1 ! PUSHL Length
;    0394  1 ! PUSHAB Buffer
;    0395  1 !    CALLS #2,L^CRD$Log_Write
;    0396  1 !
;    0397  1 !   Return Status:
;    0398  1 !    All operation, pass or fail return success
;    0399  1 !-
;    0400  1 !**************************************************************************
;    0401  2 BEGIN
;    0402  2 BUILTIN
;    0403  2     ACTUALPARAMETER;
;    0404  2
;    0405  2 ! Write the passed string into the log file
;    0406  2
;    0407  2 Log_RAB[RAB$L_FAB] = Log_FAB; ! This is for Position Independent code      ! [06]
;    0408  2 Log_RAB[RAB$L_RBF] = ACTUALPARAMETER(1);          ! [06]
;    0409  2 Log_RAB[RAB$W_RSZ] = ACTUALPARAMETER(2);         ! [06]
;    0410  2 $WRITE (RAB=Log_RAB);              ! [06]
;    0411  2 RETURN CRD$K_Success;              ! [06]
;    0412  1 END;                ! [06]



         0000 00000     .ENTRY  CRD$LOG_WRITE, Save nothing          ; 0384
    00000000'  EF 00000000'  EF  9E 00002    MOVAB   LOG_FAB, LOG_RAB+60          ; 0407
    00000000'  EF 04    AC  D0 0000D     MOVL    4(AP), LOG_RAB+40           ; 0408
    00000000'  EF 08    AC  B0 00015     MOVW    8(AP), LOG_RAB+34           ; 0409
    00000000'  EF  9F 0001D     PUSHAB  LOG_RAB             ; 0410
    00000000G  00       01  FB 00023     CALLS   #1, SYS$WRITE           ;
        50       01  D0 0002A     MOVL    #1, R0              ; 0411
     04 0002D    RET                 ; 0412

; Routine Size:  46 bytes,    Routine Base:  CODE + 0202


;    0413  1
;    0414  1 END     ! End of CRDINTRFC module
;    0415  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          136  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK          304  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          560  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$BITPOSITION   Macro    Lib04   109
$CLOSE    KeyWMacr Lib04   378
$CRD_LITERALS   Macro    Lib01    79
$CRD_RETURN_LENGTH_STATUS Macro    Lib01   285
$DS_DSADEF   Macro    Lib03    78
$DS_TYPEDEF   Macro    Lib03    80
$EQULST    Macro    Lib04    79       80
$ER    Comptime   100
$ERASE    KeyWMacr Lib04   333
$FAB    KeyWMacr Lib04   109
$FAB_DECL   Macro    Lib04   109
$FAO    Macro    Lib04   347
$LOAD_ASCID_S   KeyWMacr   102   274     343
$MODULE    Bind       100
$OPEN    KeyWMacr Lib04   339
$RAB    KeyWMacr Lib04   110
$RAB_DECL   Macro    Lib04   110
$RAB_DECL_ASYN   Unbound          110
$RMS_BITFLD   Macro    Lib04   109     110
$RMS_BITS   Unbound          109     110
    Macro    Lib04   109
$RMS_CODFLD   Macro    Lib04   109     110
$RMS_OR    Macro    Lib04   109
$RMS_VALFLD   Unbound          109
    Macro    Lib04   109     110
$WRITE    KeyWMacr Lib04   356     410
ACTUALPARAMETER   Builtin    403   408     409
ADDRESS    RoutForm   118   259.    279a=   291.
AP    Builtin    408   408.
    Builtin    409   409.
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     79
AUTO$K_EXIT_CONTROL_C  Literal     79
AUTO$K_EXIT_DUMMY_2  Literal     79
AUTO$K_EXIT_DUMMY_3  Literal     79
AUTO$K_EXIT_ERRORS_COMPLETE Literal     79
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     79
AUTO$K_EXIT_INTERNAL_ERROR Literal     79
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     79
AUTO$K_EXIT_LAST_REASON  Literal     79    79
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     79
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     79
AUTO$K_EXIT_NOERRORS_PREP Literal     79
CODE    RoutForm   118   196.    227.    237.    245.    263.    271.
COMMAND    Own        167   274=    275.    276a.
CRD$CRD_INITIALIZATION  ExtRout  Lib01   209c
CRD$DS_INTERFACE  GlobRout   118 Lib01e     69f
CRD$GB_USER_PROMPT_OKAY  Global     113 Lib01e    113=    227.
CRD$K_ACTION_EQL_DO_NOTHING Literal     79
CRD$K_ACTION_RANGE_ERROR Literal     79
CRD$K_ADVANCE_DIAGNOSTIC Literal     79
CRD$K_ADVANCE_GENERAL_COM Literal     79
CRD$K_ADVANCE_STATE  Literal     79
CRD$K_ALLOCATION_ERROR  Literal     79
CRD$K_ASSIGN_PTABLE_PTRS Literal     79
CRD$K_AUTOSIZER_ERROR  Literal     79
CRD$K_AUTOSIZER_SET_FLAGS Literal     79
CRD$K_BAD_ACTION_NUMBER  Literal     79
CRD$K_BAD_TYPECODE_ERROR Literal     79
CRD$K_BASE_SYSTEM_IDC  Literal     79
CRD$K_BASE_SYSTEM_LESI  Literal     79
CRD$K_BASE_SYSTEM_NULL  Literal     79
CRD$K_BASE_SYSTEM_UDA_0  Literal     79
CRD$K_BASE_SYSTEM_UDA_1  Literal     79
CRD$K_BASE_SYSTEM_UDA_2  Literal     79
CRD$K_BASE_SYSTEM_UDA_3  Literal     79
CRD$K_CRD_INITIALIZATION Literal     79   198
CRD$K_CREATE_HST  Literal     79
CRD$K_DATA_LENGTH_ERROR  Literal     79
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     79
CRD$K_DDB_BLINK   Literal     79
CRD$K_DDB_FLINK   Literal     79
CRD$K_DDB_LAST_ENTRY  Literal     79
CRD$K_DDB_MEMORY_SIZE  Literal     79
CRD$K_DDB_MENU_DEVICE_NUMB Literal     79
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     79
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     79
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     79
CRD$K_DDB_PTABLE_ENTRY  Literal     79
CRD$K_DDB_STATUS  Literal     79
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     79
CRD$K_DEVICE_NAME  Literal     79
CRD$K_DIAGNOSTIC_ERROR  Literal     79
CRD$K_DIAGNOSTIC_NAME  Literal     79
CRD$K_DONE_GENERAL_COMS  Literal     79
CRD$K_DO_NOTHING  Literal     79
CRD$K_DS_COMMAND_SIZE  Literal  Lib01   285
CRD$K_DUMMY_10   Literal     79
CRD$K_DUMMY_11   Literal     79
CRD$K_DUMMY_12   Literal     79
CRD$K_DUMMY_13   Literal     79
CRD$K_DUMMY_3   Literal     79
CRD$K_DUMMY_4   Literal     79
CRD$K_DUMMY_5   Literal     79
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
CRD$K_DUMMY_6   Literal     79
CRD$K_DUMMY_7   Literal     79
CRD$K_DUMMY_8   Literal     79
CRD$K_DUMMY_9   Literal     79
CRD$K_EXIT_CRD   Literal     79
CRD$K_HOOK_POINT  Literal     79   194
CRD$K_HS_INTERNAL_ERROR  Literal     79
CRD$K_IDC_EVDLB   Literal     79
CRD$K_IDC_EVDLC   Literal     79
CRD$K_IDC_EVDLD   Literal     79
CRD$K_IDC_EVRAD_0  Literal     79
CRD$K_IDC_EVRAD_1  Literal     79
CRD$K_IDC_LAST_DIAGNOSTIC Literal     79
CRD$K_INTERNAL_ERROR  Literal     79
CRD$K_LAST_ACTION_ROUTINE Literal     79
CRD$K_LAST_ENTRY  Literal     79
CRD$K_LAST_FUNCTION  Literal     79   192
CRD$K_LAST_HOOK_POINT  Literal     79   196
CRD$K_LAST_INTERNAL_ERROR Literal     79
CRD$K_LAST_TYPE   Literal     79
CRD$K_LESI_ENWCA  Literal     79
CRD$K_LESI_EVRMB_0  Literal     79
CRD$K_LESI_EVRMB_1  Literal     79
CRD$K_LESI_LAST_DIAGNOSTIC Literal     79
CRD$K_LEVEL_4_PASSED  Literal     79
CRD$K_LOAD_AUTOSIZER  Literal     79
CRD$K_LOAD_ERROR  Literal     79
CRD$K_LOAD_GENERAL_COM  Literal     79
CRD$K_LOAD_LOAD_COM  Literal     79
CRD$K_LOAD_SELECT_COM  Literal     79
CRD$K_LOAD_SET_EVENT_COM Literal     79
CRD$K_LOAD_SET_FLAGS_COM Literal     79
CRD$K_LOAD_START_COM  Literal     79
CRD$K_LOAD_SUP_FILE  Literal     79
CRD$K_MM_INTERNAL_ERROR  Literal     79
CRD$K_NEW_LINE   Literal     79
CRD$K_PREPARATION_ERROR  Literal     79
CRD$K_PREPARATION_ERROR_TEXT Literal     79
CRD$K_RUNTIME   Literal     79
CRD$K_SAME_LINE   Literal     79
CRD$K_SET_EVENT_ARGS  Literal     79
CRD$K_SET_FLAGS_ARGS  Literal     79
CRD$K_SET_UP_DIAGNOSTIC  Literal     79
CRD$K_START   Literal     79
CRD$K_START_AUTOSIZER  Literal     79
CRD$K_START_ERROR  Literal     79
CRD$K_START_QUALIFIERS  Literal     79
CRD$K_STAR_LINE   Literal     79
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal     79
CRD$K_STATE_ADVANCE_GENERAL_COM Literal     79
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal     79
CRD$K_STATE_AUTOSIZER_ERROR Literal     79
CRD$K_STATE_AUTOSIZER_RUNNING Literal     79
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------- -------- ----- ---------------
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal     79
CRD$K_STATE_DIAGNOSTIC_ERROR Literal     79
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal     79
CRD$K_STATE_DONE_DIAGNOSTICS Literal     79
CRD$K_STATE_DONE_GENERAL_COMS Literal     79
CRD$K_STATE_DUMMY_1  Literal     79
CRD$K_STATE_DUMMY_10  Literal     79
CRD$K_STATE_DUMMY_12  Literal     79
CRD$K_STATE_DUMMY_13  Literal     79
CRD$K_STATE_DUMMY_2  Literal     79
CRD$K_STATE_DUMMY_3  Literal     79
CRD$K_STATE_DUMMY_4  Literal     79
CRD$K_STATE_DUMMY_6  Literal     79
CRD$K_STATE_DUMMY_7  Literal     79
CRD$K_STATE_DUMMY_8  Literal     79
CRD$K_STATE_EXIT_CRD  Literal     79
CRD$K_STATE_INTERNAL_ERROR Literal     79
CRD$K_STATE_LAST_STATE  Literal     79
CRD$K_STATE_LEVEL_4_PASSED Literal     79
CRD$K_STATE_LOAD_AUTOSIZER Literal     79
CRD$K_STATE_LOAD_ERROR  Literal     79
CRD$K_STATE_LOAD_GENERAL_COM Literal     79
CRD$K_STATE_LOAD_LOAD_COM Literal     79
CRD$K_STATE_LOAD_SELECT_COM Literal     79
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     79
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     79
CRD$K_STATE_LOAD_START_COM Literal     79
CRD$K_STATE_PREPARATION_ERROR Literal     79
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     79
CRD$K_STATE_START  Literal     79
CRD$K_STATE_START_AUTOSIZER Literal     79
CRD$K_STATE_START_ERROR  Literal     79
CRD$K_SUCCESS   Unbound          285
    Literal  Lib01   264     285     357     380     411
CRD$K_TEST_START_TEXT  Literal     79
CRD$K_TQ_INTERNAL_ERROR  Literal     79
CRD$K_TYPECODE   Literal     79   221
CRD$K_UDA_0_EVDLB  Literal     79
CRD$K_UDA_0_EVDLC  Literal     79
CRD$K_UDA_0_EVDLD  Literal     79
CRD$K_UDA_0_EVRLA_0  Literal     79
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     79
CRD$K_UDA_1_EVDLB  Literal     79
CRD$K_UDA_1_EVDLC  Literal     79
CRD$K_UDA_1_EVDLD  Literal     79
CRD$K_UDA_1_EVRLA_0  Literal     79
CRD$K_UDA_1_EVRLA_1  Literal     79
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     79
CRD$K_UDA_2_EVDLB  Literal     79
CRD$K_UDA_2_EVDLC  Literal     79
CRD$K_UDA_2_EVDLD  Literal     79
CRD$K_UDA_2_EVRLA_0  Literal     79
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     79
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_UDA_3_EVDLB  Literal     79
CRD$K_UDA_3_EVDLC  Literal     79
CRD$K_UDA_3_EVDLD  Literal     79
CRD$K_UDA_3_EVRLA_0  Literal     79
CRD$K_UDA_3_EVRLA_1  Literal     79
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     79
CRD$LOG_START   GlobRout   303    72f
CRD$LOG_STOP   GlobRout   360    73f
CRD$LOG_WRITE   GlobRout   384    74f
CRD$TURING_MACHINE  ExtRout  Lib01   259c
DESC    Own         321   346=    350a
DS$K_PRINTB   Literal     80
DS$K_PRINTF   Literal     80
DS$K_PRINTI   Literal     80
DS$K_PRINTX   Literal     80
DS$K_TYPE_ABORT_PROGRAM  Literal     80
DS$K_TYPE_ABORT_TEST  Literal     80
DS$K_TYPE_COMMAND_ERR  Literal     80
DS$K_TYPE_COMMAND_OUT  Literal     80
DS$K_TYPE_CRD_AUTOTEST  Literal     80
DS$K_TYPE_DS_PROMPT  Literal     80   240     271
DS$K_TYPE_DS_START  Literal     80   263
DS$K_TYPE_ERRDEV  Literal     80
DS$K_TYPE_ERRHARD  Literal     80
DS$K_TYPE_ERROR_BODY  Literal     80
DS$K_TYPE_ERROR_END  Literal     80
DS$K_TYPE_ERRPREP  Literal     80
DS$K_TYPE_ERRSOFT  Literal     80
DS$K_TYPE_ERRSUP  Literal     80
DS$K_TYPE_ERRSYS  Literal     80
DS$K_TYPE_ERR_HALT  Literal     80
DS$K_TYPE_EXCEPTION  Literal     80
DS$K_TYPE_EXCEPTION_HEAD Literal     80
DS$K_TYPE_FIRST_PASS  Literal     80
DS$K_TYPE_GENERAL  Literal     80
DS$K_TYPE_GENERAL_ERROR  Literal     80
DS$K_TYPE_NO_TESTS  Literal     80
DS$K_TYPE_PARAM_ERROR  Literal     80
DS$K_TYPE_PROGRAM_END  Literal     80
DS$K_TYPE_PROGRAM_INFO  Literal     80
DS$K_TYPE_PROGRAM_START  Literal     80
DS$K_TYPE_QIO_INVADP  Literal     80
DS$K_TYPE_QIO_NODRIVER  Literal     80
DS$K_TYPE_QIO_WRONGVER  Literal     80
DS$K_TYPE_SCRIPT_ECHO  Literal     80   237
DS$K_TYPE_SCRIPT_PNF  Literal     80
DS$K_TYPE_SCRIPT_PROMPT  Literal     80
DS$K_TYPE_SCRIPT_SKIP  Literal     80
DS$K_TYPE_SEQUENCE_ERROR Literal     80
DS$K_TYPE_START_ERR  Literal     80
DS$K_TYPE_START_LIST  Literal     80
DS$K_TYPE_SUMMARY  Literal     80
DS$K_TYPE_USER_PROMPT  Literal     80   227
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DSA$AL_APTMAIL    Bind          78     78
DSA$GL_APTCOM    Bind           78
DSA$GL_DEVLEN    Bind           78
DSA$GL_DEVNAM    Bind           78
DSA$GL_ERRNO    Bind            78
DSA$GL_EVENT    Bind            78
DSA$GL_FLAGS    Bind            78   182      183      184      281      282      283      284
DSA$GL_MSGPTR    Bind           78
DSA$GL_MSGTYP    Bind           78
DSA$GL_PASSES    Bind           78
DSA$GL_PASSNO    Bind           78
DSA$GL_SECTNO    Bind           78
DSA$GL_SID    Bind          78    208      258      262      269      270
DSA$GL_SUBTNO    Bind           78
DSA$GL_TESTNO    Bind           78
DSA$GL_UNITS    Bind           78
DSA$V_CRD_AUTOTEST_OFF  Macro    Lib03   182      281
DSA$V_CRD_MENUTEST_OFF  Macro    Lib03   183      282
DSA$V_CRD_MENUTEST_ON  Macro    Lib03   184      283
DSA$V_CRD_TRACE    Macro    Lib03   284
DSA_EXTRACT    Register   164   181=    182=    183=    184=    203.    213.    252.
DSC$K_CLASS_S    Unbound          105
    Literal  Lib04   274      346
DSC$K_DTYPE_T    Unbound          104
    Literal  Lib04   274      346
FAB$B_FNS    Macro    Lib04   332      338
FAB$C_BID    Literal  Lib04   109
FAB$C_BLN    Literal  Lib04   109
FAB$C_SEQ    Literal  Lib04   109
FAB$C_VAR    Literal  Lib04   109
FAB$L_FNA    Macro    Lib04   331      337
FAB$M_CR    Literal  Lib04   109
FAB$M_GET    Literal  Lib04   109
FAB$V_CHAN_MODE    Macro    Lib04   109
FAB$V_FILE_MODE    Macro    Lib04   109
FAB$V_LNM_MODE    Macro    Lib04   109
FALSE    Literal  Lib01   113
FUNCTION    RoutForm   118   192.
GET1ST_    Macro    Lib04    79       80
GET2ND_    Unbound           79       80
HEADER_BUFF    Own          323   350a    354.
HEADER_SIZE    Own          322   350a    355.
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal      79
HS$K_ACTION_ADVANCE_GENERAL_COM Literal      79
HS$K_ACTION_ADVANCE_STATE Literal      79
HS$K_ACTION_CHANGE_TABLE Literal      79
HS$K_ACTION_DIAGNOSTIC_ERROR Literal      79
HS$K_ACTION_DONE_GENERAL_COMS Literal      79
HS$K_ACTION_DO_NOTHING  Literal      79
HS$K_ACTION_DUMMY_3  Literal      79
HS$K_ACTION_DUMMY_4  Literal      79
HS$K_ACTION_DUMMY_5  Literal      79
HS$K_ACTION_DUMMY_6  Literal      79
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_ACTION_INIT_HS  Literal     79
HS$K_ACTION_INTERNAL_ERROR Literal     79
HS$K_ACTION_LAST_ACTION  Literal     79
HS$K_ACTION_LOAD_ERROR  Literal     79
HS$K_ACTION_LOAD_GENERAL_COM Literal     79
HS$K_ACTION_LOAD_LOAD_COM Literal     79
HS$K_ACTION_LOAD_SELECT_COM Literal     79
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     79
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     79
HS$K_ACTION_LOAD_START_COM Literal     79
HS$K_ACTION_PREPARATION_ERROR Literal     79
HS$K_ACTION_START_ERROR  Literal     79
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     79
HS$K_STATE_ADVANCE_GENERAL_COM Literal     79
HS$K_STATE_CHANGE_TABLE  Literal     79
HS$K_STATE_DIAGNOSTIC_ERROR Literal     79
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     79
HS$K_STATE_DONE_GENERAL_COMS Literal     79
HS$K_STATE_DUMMY_1  Literal     79
HS$K_STATE_DUMMY_2  Literal     79
HS$K_STATE_DUMMY_3  Literal     79
HS$K_STATE_DUMMY_4  Literal     79
HS$K_STATE_DUMMY_6  Literal     79
HS$K_STATE_INIT_HS  Literal     79
HS$K_STATE_INTERNAL_ERROR Literal     79
HS$K_STATE_LAST_STATE  Literal     79
HS$K_STATE_LOAD_ERROR  Literal     79
HS$K_STATE_LOAD_GENERAL_COM Literal     79
HS$K_STATE_LOAD_LOAD_COM Literal     79
HS$K_STATE_LOAD_SELECT_COM Literal     79
HS$K_STATE_LOAD_SET_EVENT_COM Literal     79
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     79
HS$K_STATE_LOAD_START_COM Literal     79
HS$K_STATE_PREPARATION_ERROR Literal     79
HS$K_STATE_START_ERROR  Literal     79
LENGTH    RoutForm   118   246.
LOG_FAB    Own        109   109=    327a    331=    332=    333a    337=    338=    339a    377a    378a
            407a
LOG_RAB    Own        110   110=    327=    354=    355=    356a    377=    407=    408=    409-    410a
MEN$K_HS_STATE_TABLE  Literal     79
MEN$K_MM_STATE_TABLE  Literal     79
MEN$K_TQ_STATE_TABLE  Literal     79
MEN$MENU_FNCTST_INIT  ExtRout  Lib01   213c
MEN$TURING_MACHINE  ExtRout  Lib01   291c
MENU$K_EXIT_AUTOSIZER_ERROR Literal     79
MENU$K_EXIT_INTERNAL_ERROR Literal     79
MENU$K_EXIT_LAST_REASON  Literal     79
MENU$K_EXIT_OPERATOR_ABORT Literal     79
MENU$K_EXIT_OPERATOR_STOP Literal     79
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     79
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     79
MM$K_ACTION_ADVANCE_STATE Literal     79
MM$K_ACTION_AUTOSIZER_ERROR Literal     79
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_ACTION_BUILD_DDBS  Literal     79
MM$K_ACTION_BUILD_HSTS  Literal     79
MM$K_ACTION_CHANGE_TABLE Literal     79
MM$K_ACTION_DONE_INITS  Literal     79
MM$K_ACTION_DO_NOTHING  Literal     79
MM$K_ACTION_DUMMY_3  Literal     79
MM$K_ACTION_DUMMY_4  Literal     79
MM$K_ACTION_DUMMY_5  Literal     79
MM$K_ACTION_DUMMY_6  Literal     79
MM$K_ACTION_DUMMY_7  Literal     79
MM$K_ACTION_DUMMY_8  Literal     79
MM$K_ACTION_INTERNAL_ERROR Literal     79
MM$K_ACTION_LAST_ACTION  Literal     79
MM$K_ACTION_LOAD_AUTOSIZER Literal     79
MM$K_ACTION_MENU_PROMPT  Literal     79
MM$K_ACTION_PRINT_MENU  Literal     79
MM$K_ACTION_PRINT_MENU_HEADING Literal     79
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     79
MM$K_ACTION_START_AUTOSIZER Literal     79
MM$K_STATE_AUTOSIZER_ERROR Literal     79
MM$K_STATE_AUTOSIZER_RUNNING Literal     79
MM$K_STATE_BUILD_DDBS  Literal     79
MM$K_STATE_BUILD_HSTS  Literal     79
MM$K_STATE_CHANGE_TABLE  Literal     79
MM$K_STATE_DONE_INITS  Literal     79
MM$K_STATE_DUMMY_1  Literal     79
MM$K_STATE_DUMMY_2  Literal     79
MM$K_STATE_DUMMY_3  Literal     79
MM$K_STATE_DUMMY_5  Literal     79
MM$K_STATE_DUMMY_7  Literal     79
MM$K_STATE_DUMMY_8  Literal     79
MM$K_STATE_INTERNAL_ERROR Literal     79
MM$K_STATE_LAST_STATE  Literal     79
MM$K_STATE_LOAD_AUTOSIZER Literal     79
MM$K_STATE_MENU_PROMPT  Literal     79
MM$K_STATE_PRINT_MENU  Literal     79
MM$K_STATE_PRINT_MENU_HEADING Literal     79
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     79
MM$K_STATE_START  Literal     79
MM$K_STATE_START_AUTOSIZER Literal     79
MODNAM    Macro    Lib02   100
NAM    KeyWForm   102   103     104     105     106
NUL2ND     Macro    Lib04    79      80
P1     Register   164   240=    245=    259.    291.
P2     Register   164   241=    246=    259.    278.    291.
PR$_SID_TYP730   Literal  Lib04   208     258
PR$_SID_TYP8NN   Literal  Lib04   270
PR$_SID_TYP8SS   Literal  Lib04   262     269
RAB$C_BID   Literal  Lib04   110
RAB$C_BLN   Literal  Lib04   110
RAB$C_SEQ   Literal  Lib04   110
RAB$L_FAB   Macro    Lib04   327     377     407
RAB$L_RBF   Macro    Lib04   354     408
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
RAB$W_RSZ    Macro    Lib04   355      409
RETURN_VALUE    Register   164   209=    210=    213=    214=    215=    218=    229=    259=    264=    287=
              291=    292=    293=    296=    298.
SDL$STARLET_CONCAT  Macro    Lib04   333      339      356      378      410
SDL$STARLET_OPT    Macro    Lib04   333      339      356      378      410
SDL$STARLET_REQ    Macro    Lib04   333      339      356      378      410
STR    KeyWForm    102   103      106
SYS$CLOSE    ExtRout    378   378c
SYS$ERASE    ExtRout    333   333c
SYS$FAO    ExtRout    350   350c
SYS$OPEN    ExtRout    339   339c
SYS$WRITE    ExtRout    356   356c    410e    410c
TQ$K_ACTION_ADVANCE_STATE Literal     79
TQ$K_ACTION_CHANGE_TABLE Literal     79
TQ$K_ACTION_DONE_TEST_QUEUE Literal     79
TQ$K_ACTION_DO_NOTHING  Literal     79
TQ$K_ACTION_DUMMY_3  Literal     79
TQ$K_ACTION_DUMMY_4  Literal     79
TQ$K_ACTION_DUMMY_5  Literal     79
TQ$K_ACTION_GET_DDB  Literal     79
TQ$K_ACTION_INIT_TQ  Literal     79
TQ$K_ACTION_INTERNAL_ERROR Literal     79
TQ$K_ACTION_LAST_ACTION  Literal     79
TQ$K_STATE_CHANGE_TABLE  Literal     79
TQ$K_STATE_DONE_TEST_QUEUE Literal     79
TQ$K_STATE_DUMMY_1  Literal     79
TQ$K_STATE_DUMMY_2  Literal     79
TQ$K_STATE_DUMMY_3  Literal     79
TQ$K_STATE_DUMMY_4  Literal     79
TQ$K_STATE_DUMMY_5  Literal     79
TQ$K_STATE_GET_DDB  Literal     79
TQ$K_STATE_INIT_TQ  Literal     79
TQ$K_STATE_INTERNAL_ERROR Literal     79
TQ$K_STATE_LAST_STATE  Literal     79


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDINTRFC.B32;1
    2  Module CRDINTRFC
   59  Library #1 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   60  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   61  Library #3 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   62  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  415  Eludom CRDINTRFC
```

```
 KEY TO REFERENCE TYPE FLAGS

      . Fetch
      = Store
      c Routine call
      a Address use
      a Indirect use
      e External, external routine, or external literal declaration
      f Forward or forward routine declaration
      h Condition handler enabling
      m Map declaration
      u Undeclare declaration


;     Library Statistics
;
;             -------- Symbols --------     Pages Processing
; File            Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;              486         11          2          62          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;              671          1          0          46          00:00.2
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;              923          6          0          94          00:00.2
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619         43           0       1000          00:04.5



;      COMMAND QUALIFIERS

; BLISS CRDINTRFC.B32/LIST=CRDINTRFC.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDINTRFC.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  560 code + 440 data bytes
; Run Time:     00:26.0
; Elapsed Time:     00:48.5
; Lines/CPU Min:      956
; Lexemes/CPU-Min: 55274
; Memory Used: 224 pages
; Compilation Complete
```

```
;    0001  0 %TITLE '*** CRDMMACT Module'
;    0002  0 MODULE CRDMMACT (    ! Action routines for the Main Menu State Table
;    0003  0    IDENT = 'V01.4'
;    0004  0    ) =
;    0005  0 !++
;    0006  0 ! COPYRIGHT (c) 1980, 1981,1985
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! This module contains those action routines for the Main Menu
;    0032  0 ! State Table.
;    0033  0 !
;    0034  0 ! History:
;    0035  0 !
;    0036  0 ! 00 Jack Stansbury, July 30,1982
;    0037  0 !  Created module
;    0038  0 !
;    0039  0 ! 01 Jack Stansbury, March 3, 1983, Version ?.?
;    0040  0 !  Changed the debugging strategy - i.e., the strategy dealing with the CRD Debug Vector.
;    0041  0 !  Took all the code out of the
;    0042  0 !
;    0043  0 ! 02 Scott Cohen, June 17, 1983, Version 1.2
;    0044  0 !  Added support for soft console by
;    0045  0 !  - $SCREEN_Literals.
;    0046  0 !  - Changed MM$Build_DDBs to pause when AutoSizer is finished.
;    0047  0 !
;    0048  0 ! 03 Ron Parker October 17,1984
;    0049  0 !  - Modified Load, Set_flags and Start autosizer routines
;    0050  0 !    for the online autosizer.
;    0051  0 !
;    0052  0 ! 04 Ron Parker Febuary 27,1985
;    0053  0 !  - Modified load, !    for the online autosizer.
;    0054  0 !--
```

```
;    0055  0 %SBTTL 'Libraries'
;    0056  0
;    0057  0
;    0058  1 BEGIN      ! Begin the CRDMMACT module
;    0059  1
;    0060  1 LIBRARY
;    0061  1   '$DS';     ! Use Ds.L32 f'rst
;    0062  1
;    0063  1 LIBRARY
;    0064  1   '$Diag';    ! Use Diag.L32 second
;    0065  1
;    0066  1 LIBRARY
;    0067  1   '$CRD';     ! Use CRD.L32 third
;    0068  1
;    0069  1 LIBRARY
;    0070  1   'DISK$DS:[DS.EVSBB.DEF]EVSBB_DEF';! Use Lib for EVSBB stuff
;    0071  1
;    0072  1 LIBRARY
;    0073  1   'Sys$Library:Lib';  ! Use Lib as last resort
;    0074  1
;    0075  1
;    0076  1
;    0077  1 %SBTTL 'Table of Contents'
;    0078  1
;    0079  1 EXTERNAL ROUTINE
;    0080  1   MEN$GENERATE_ONLINE_ATTACH : ADDRESSING_MODE (LONG_RELATIVE);
;    0081  1
;    0082  1 FORWARD ROUTINE
;    0083  1   MM$Print_Menu_Heading  : ADDRESSING_MODE (LONG_RELATIVE),
;    0084  1   MM$Load_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE),
;    0085  1   MM$Set_Flags_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE),
;    0086  1   MM$Start_AutoSizer  : ADDRESSING_MODE (LONG_RELATIVE),
;    0087  1
;    0088  1   MM$Build_DDBs    : ADDRESSING_MODE (LONG_RELATIVE),
;    0089  1   MM$Build_HSTs    : ADDRESSING_MODE (LONG_RELATIVE),
;    0090  1
;    0091  1   MM$Done_Inits    : ADDRESSING_MODE (LONG_RELATIVE),
;    0092  1
;    0093  1   MM$Print_Menu    : ADDRESSING_MODE (LONG_RELATIVE),
;    0094  1   MM$Change_Table    : ADDRESSING_MODE (LONG_RELATIVE),
;    0095  1   MM$Menu_Prompt    : ADDRESSING_MODE (LONG_RELATIVE),
;    0096  1
;    0097  1   MM$Internal_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    0098  1   MM$AutoSizer_Error  : ADDRESSING_MODE (LONG_RELATIVE),
;    0099  1
;    0100  1   MEN$Menu_Test_Internal_Error : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
;    0101  1
;    0102  1
;    0103  1
;    0104  1 %SBTTL 'Included Macros and Literals'
;    0105  1
;    0106  1   $DS_DSADef;    ! The DSA flags and locations
;    0107  1   $CRD_Literals;    ! CRD Literals
;    0108  1   $SCREEN_Literals;  ! Define the SCREEN operation literals.
;    0109  1
```

```
;    0110  1 %SBTTL 'Psect Definitions'
;    0111  1
;    0112  1 PSECT
;    0113  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0114  1
;    0115  1 PSECT
;    0116  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0117  1
;    0118  1 PSECT
;    0119  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0120  1
;    0121  1 PSECT
;    0122  1     GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0123  1
;    0124  1
;    0125  1
;    0126  1 %SBTTL 'Bindings'
;    0127  1
;    0128  1  ModNam ('CRDMMACT');  ! Module name for ErrSup macro
;    0129  1
;    0130  1 GLOBAL BIND
;    0131  1     CRD$GT_AutoSizer_Name      = $ASCIC ('EVSBA'),
;    0132  1     CRD$GT_AutoSizer_Set_Flags = $ASCIC ('QUICK');
;    0133  1
;    0134  1
;    0135  1 %SBTTL 'VMS Online Autosizer Error Handling macro'
;    0136  1
; M  0137  1 MACRO $Check_OA_Status ( Arg ) =     ! [05]
; M  0138  1     IF (NOT .Arg) THEN       ! [05]
; M  0139  1  BEGIN        ! [05]
; M  0140  1  IF (.DSA$V_CRD_Trace) THEN     ! [05]
; M  0141  1      $CRD_PRINT (      ! [05]
; M  0142  1   $ASCIC ('*** ERROR - !AC = !XL'),   ! [05]
; M  0143  1   $ASCIC (%QUOTENAME(Arg)),    ! [05]
; M  0144  1   .Arg);       ! [05]
; M  0145  1          $CRD_Print (MEN$GT_Failed_ONL_Autosizer);   ! [05]
; M  0146  1  $CRD_Print (CRD$GT_Autosizer_Error);    ! [05]
; M  0147  1  CRD$STOP (MENU$K_Exit_Autosizer_Error);    ! [05]
; M  0148  1  RETURN (CRD$K_Failure);      ! [05]
;    0149  1          END %;
;    0150  1
;    0151  1 %SBTTL 'Position Independant ASCID macro'
;    0152  1
; M  0153  1 KEYWORDMACRO $LOAD_ASCID_S ( NAM, STR ) =
; M  0154  1     BEGIN
; M  0155  1     NAM[0,00,16,0] = %CHARCOUNT(STR);     ! [03]
; M  0156  1     NAM[0,16,08,0] = DSC$K_DTYPE_T;      ! [03]
; M  0157  1     NAM[0,24,08,0] = DSC$K_CLASS_S;     ! [03]
; M  0158  1     NAM[1,00,32,0] = UPLIT (%ASCII STR);    ! [03]
;    0159  1     END%;
;    0160  1
; M  0161  1 KEYWORDMACRO $LOAD_ASCID_A ( NAM, ADR, SIZ=0 ) =
; M  0162  1     BEGIN
; M  0163  1     NAM[0,00,16,0] = SIZ;      ! [04]
; M  0164  1     NAM[0,16,08,0] = DSC$K_DTYPE_T;      ! [03]
```

```
; M 0165  1     NAM[0,24,08,0] = DSC$K_CLASS_S;     ! [03]
; M 0166  1     NAM[1,00,32,0] = ADR;       ! [04]
;   0167  1     ENDx;
```

```
;    0168  1 %SBTTL 'MM$Print_Menu_Heading Routine'
;    0169  1
;    0170  1 GLOBAL ROUTINE MM$Print_Menu_Heading =
;    0171  1
;    0172  1 !++
;    0173  1 ! Functional Description:
;    0174  1 !
;    0175  1 ! Describe function of routine
;    0176  1 !
;    0177  1 ! Formal Input Parameters:
;    0178  1 !
;    0179  1 ! None
;    0180  1 !
;    0181  1 ! Formal Output Parameters:
;    0182  1 !
;    0183  1 ! None
;    0184  1 !
;    0185  1 ! Side Effects:
;    0186  1 !
;    0187  1 ! None
;    0188  1 !
;    0189  1 ! Completion Codes:
;    0190  1 !
;    0191  1 ! None
;    0192  1 !--
```

```
;   0193  2 BEGIN     ! Routine MM$Print_Menu_Heading
;   0194  2  $CRD_Print (MEN$GT_MenuTest_Begin);
;   0195  2       ! Print the Menu Test heading
;   0196  3  RETURN (CRD$K_Repeat_Turing)
;   0197  1 END;      ! Routine MM$Print_Menu_Heading


          .TITLE  CRDMMACT *** CRDMMACT Module
          .IDENT  \V01.4\

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

   54  43  41  4D  4D  44  52  43  08  00000 P.AAA:  .ASCII  <8>\CRDMMACT\          ;
       41  42  53  56  45  05  00009 P.AAB:  .ASCII  <5>\EVSBA\           ;
       4B  43  49  55  51  05  0000F P.AAC:  .ASCII  <5>\QUICK\           ;

        DSA$AL_APTMAIL=     65024
        DSA$GL_FLAGS=       65024
        DSA$GL_APTCOM=      65028
        DSA$GL_PASSES=      65032
        DSA$GL_UNITS=       65036
        DSA$GL_SECTNO=      65040
        DSA$GL_SID=       65044
        DSA$GL_MSGTYP=      65088
        DSA$GL_ERRNO=       65092
        DSA$GL_EVENT=       65096
        DSA$GL_SUBTNO=      65100
        DSA$GL_TESTNO=      65104
        DSA$GL_PASSNO=      65108
        DSA$GL_DEVLEN=      65112
        DSA$GL_DEVNAM=      65116
        DSA$GL_MSGPTR=      65128
        $MODULE=        P.AAA
        CRD$GT_AUTOSIZER_NAME==
              P.AAB
        CRD$GT_AUTOSIZER_SET_FLAGS==
              P.AAC
          .EXTRN  MEN$GENERATE_ONLINE_ATTACH
          .EXTRN  MM$PRINT_MENU_HEADING
          .EXTRN  MM$LOAD_AUTOSIZER
          .EXTRN  MM$SET_FLAGS_AUTOSIZER
          .EXTRN  MM$START_AUTOSIZER
          .EXTRN  MM$BUILD_DDBS, MM$BUILD_HSTS
          .EXTRN  MM$DONE_INITS, MM$PRINT_MENU
          .EXTRN  MM$CHANGE_TABLE
          .EXTRN  MM$MENU_PROMPT, MM$INTERNAL_ERROR
          .EXTRN  MM$AUTOSIZER_ERROR
          .EXTRN  MEN$MENU_TEST_INTERNAL_ERROR
          .EXTRN  MEN$GT_FAILED_ONL_AUTOSIZER
          .EXTRN  CRD$GT_AUTOSIZER_ERROR
          .EXTRN  CRD$STOP, MEN$GT_MENUTEST_BEGIN
          .EXTRN  CRD$PRINT

          .PSECT  CODE,NOWRT,  SHR,  PIC,2
```

```
                0000 00000     .ENTRY  MM$PRINT_MENU_HEADING, Save nothing       ; 0170
        000C  00G  EF  9F 00002     PUSHAB  MEN$GT_MENUTEST_BEGIN          ; 0194
              01  DD 00008     PUSHL   #1              ;
              1A  DD 0000A     PUSHL   #26             ;
        00000000G  FF    03  FB 0000C     CALLS   #3, @CRD$PRINT          ;
              50    02  D0 00013     MOVL    #2, R0            ; 0196
         04 00016     RET                 ; 0197

; Routine Size:  23 bytes,    Routine Base:  CODE + 0000
```

```
;    0198  1 %SBTTL 'MM$Load_AutoSizer Routine'
;    0199  1
;    0200  1 GLOBAL ROUTINE MM$Load_Autosizer (CLI_Buffer_Length, CLI_Buffer_address) =
;    0201  1
;    0202  1 !**************************************************************************
;    0203  1 !++
;    0204  1 ! Functional Description:
;    0205  1 !
;    0206  1 ! Offline - Creates the LOAD EVSBA command in the VDS and exits
;    0207  1 !    the routine causing the command to be executed by VDS.
;    0208  1 !
;    0209  1 ! Online  - This routine will spawn a subprocess to run the ONLINE
;    0210  1 !    AUTOSIZER product. The ONLINE AUTOSIZER program will
;    0211  1 !    create a file that will be read by the Set Flags and\
;    0212  1 !    Start Autosizer routines.
;    0213  1 !
;    0214  1 !    The data in the file is a device map of the current
;    0215  1 !    VAX system.
;    0216  1 !
;    0217  1 ! There is no need to check other DSA bits for correctness as this
;    0218  1 ! has already been done in the CRD$INTERFACE routine.
;    0219  1 !
;    0220  1 ! Completion Codes:
;    0221  1 !
;    0222  1 ! (Byte_count*10000h) + CRD$K_Success
;    0223  1 ! CRD$K_Failure
;    0224  1 ! CRD$K_Success
;    0225  1 !--
;    0226  1 !**************************************************************************
```

```
;    0227  2 BEGIN     ! Routine MM$Load_AutoSizer
;    0228  2
;    0229  2 !**************************************************************************
;    0230  2 !+
;    0231  2 ! Definitions and related things
;    0232  2 !-
;    0233  2 !**************************************************************************
;    0234  2
;    0235  2 EXTERNAL ROUTINE
;    0236  2  LIB$SPAWN : ADDRESSING_MODE (LONG_RELATIVE);
;    0237  2
;    0238  2 OWN
;    0239  2    CMD : BLOCK [2,LONG],
;    0240  2    OUT : BLOCK [2,LONG],
;    0241  2    IOStatus, Status : LONG;
;    0242  2
;    0243  2 MAP
;    0244  2     CRD$GT_AutoSizer_Name : VECTOR [, BYTE],
;    0245  2     CRD$GT_DS_LOAD_Com    : VECTOR [, BYTE];
;    0246  2
;    0247  2 !**************************************************************************
;    0248  2 !+
;    0249  2 ! Start of executable statements
;    0250  2 !-
;    0251  2 !**************************************************************************
;    0252  2
;    0253  2 $CRD_Print (MEN$GT_AutoSizer_Begin);
;    0254  2
;    0255  2 !**************************************************************************
;    0256  2 !+
;    0257  2 ! IF (Menutest Offline mode) THEN
;    0258  2 !    Set up VDS to execute LOAD EVSBA command
;    0259  2 !    Return length of string & success indicator
;    0260  2 !-
;    0261  2 !**************************************************************************
;    0262  2
;    0263  3 IF (.DSA$V_CRD_Menutest_Off)      ! [03]
;    0264  2 THEN           ! [03]
;    0265  3     BEGIN         ! [03]
;    0266  3     CH$COPY (
;    0267  3  .CRD$GT_DS_LOAD_Com [0],    CH$PTR (CRD$GT_DS_LOAD_Com [1]),
;    0268  3  .CRD$GT_AutoSizer_Name [0], CH$PTR (CRD$GT_AutoSizer_Name [1]),
;    0269  3  %C' ',
;    0270  3  .CLI_Buffer_Length,  CH$PTR (.CLI_Buffer_Address)
;    0271  3  );
;    0272  3     CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0273  3     $CRD_Return_Length_Status (CRD$K_Success);
;    0274  2     END;          ! [03]
;    0275  2
```

```
;    0276  2 !**************************************************************************
;    0277  2 !+
;    0278  2 ! IF (MenuTest_Online mode) THEN
;    0279  2 !   Spawn a subprocess to run the ONLINE_AUTOSIZER, result will be a
;    0280  2 !   file that will be read by the Start Autosizer routine.
;    0281  2 !-
;    0282  2 !**************************************************************************
;    0283  2
;    0284  3 IF (.DSA$V_CRD_Menutest_On)      ! [03]
;    0285  2 THEN          ! [03]
;    0286  3     BEGIN         ! [03]
;    0287  3
;    0288  3 !   Set up PIC string descritors      ! [03]
;    0289  3     $LOAD_ASCID_S ( NAM=OUT, STR='NL:' );     ! [03]
;    0290  3     $LOAD_ASCID_S ( NAM=CMD, STR='$ RUN SYS$SYSTEM:EVSBB' );  ! [03]
;    0291  3
;    0292  3 !   Spawn a subprocess to execute the ONLINE AUTOSIZER    ! [03]
;    0293  3     IOStatus = LIB$SPAWN ( CMD,0,OUT,0,0,0,Status,0,0,0);  ! [03]
;    0294  3     $Check_OA_status (IOStatus);      ! [03]
;    0295  3     $Check_OA_status (Status);       ! [03]
;    0296  2     END;         ! [03]
;    0297  2
;    0298  2 Return CRD$K_Success;   ! Always return success  ! [03]
;    0299  1 END;     ! Routine MM$Load_AutoSizer



          .PSECT  WORK,NOEXE,2

          00000 CMD:    .BLKB   8
          00008 OUT:    .BLKB   8
          00010 IOSTATUS:
          .BLKB   4
          00014 STATUS: .BLKB   4

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

          00015    .BLKB   3
        00  3A  4C  4E  00018 P.AAD:  .ASCII  \NL:\<0>          ;
45  54  53  59  53  24  53  59  53  20  4E  55  52  20  24  0001C P.AAE:  .ASCII  \$ RUN SYS$SYSTEM:EVSBB\<0><0>       ;
  00  00  42  42  53  56  45  3A  4D  0002B                    ;
41  21  20  2D  20  52  4F  52  52  45  20  2A  2A  2A  15  00034 P.AAF:  .ASCII  <21>\*** ERROR - !AC = !XL\        ;
  4C  58  21  20  3D  20  43  00043                ;
  53  55  54  41  54  53  4F  49  08  0004A P.AAG:  .ASCII  <8>\IOSTATUS\             ;
41  21  20  2D  20  52  4F  52  52  45  20  2A  2A  2A  15  00053 P.AAH:  .ASCII  <21>\*** ERROR - !AC = !XL\        ;
  4C  58  21  20  3D  20  43  00062                ;
  53  55  54  41  54  53  06  00069 P.AAI:  .ASCII  <6>\STATUS\           ;

          .EXTRN  LIB$SPAWN, CRD$GT_DS_LOAD_COM
          .EXTRN  MEN$GT_AUTOSIZER_BEGIN
          .EXTRN  CRD$PRINT_DS_COMMAND

          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            03FC 00000    .ENTRY  MM$LOAD_AUTOSIZER, Save R2,R3,R4,R5,R6,R7,- ; 0200
```

```
          R8,R9          ;
00000000G  EF  9F 00002    PUSHAB  MEN$GT_AUTOSIZER_BEGIN        ; 0253
    01  DD 00008    PUSHL   #1              ;
    1A  DD 0000A    PUSHL   #26             ;
00000000G  FF      03  FB 0000C    CALLS   #3, @CRD$PRINT          ;
    45 0000FE02   9F  E9 00013    BLBC    a#^X0000FE02, 2$          ; 0263
    59 00000000G  EF  9A 0001A    MOVZBL  CRD$GT_DS_LOAD_COM, R9        ; 0267
    58 00000000'  EF  9A 00021    MOVZBL  CRD$GT_AUTOSIZER_NAME, R8       ; 0268
    57 04   AC  D0 00028    MOVL    CLI_BUFFER_LENGTH, R7         ; 0270
    56 08   AC  D0 0002C    MOVL    CLI_BUFFER_ADDRESS, R6          ;
57          20 00000000G  EF      59  2C 00030    MOVC5   R9, CRD$GT_DS_LOAD_COM+1, #32, R7, (R6)     ;
    66      00039          ;
    10  18 0003A    BGEQ    1$              ;
    56      59  C0 0003C    ADDL2   R9, R6          ;
    57      59  C2 0003F    SUBL2   R9, R7          ;
57          20 00000000'  EF      58  2C 00042    MOVC5   R8, CRD$GT_AUTOSIZER_NAME+1, #32, R7, (R6)  ;
    66      0004B          ;
    7E 04   AC  7D 0004C 1$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)        ; 0272
00000000G  EF      02  FB 00050    CALLS   #2, CRD$PRINT_DS_COMMAND        ;
    50 00500001   8F  D0 00057    MOVL    #5242881, R0            ; 0273
 04 0005E    RET                ;
    03 0000FE02   9F      02  E0 0005F 2$:   BBS     #2, a#^X0000FE02, 3$          ; 0284
  00D2  31 00067    BRW     7$              ;
00000000'  EF 010E0003   8F  D0 0006A 3$:   MOVL    #17694723, OUT          ; 0289
00000000'  EF 00000000'  EF  9E 00075    MOVAB   P.AAD, OUT+4            ;
00000000'  EF 010E0016   8F  D0 00080    MOVL    #17694742, CMD          ; 0290
00000000'  EF 00000000'  EF  9E 0008B    MOVAB   P.AAE, CMD+4            ;
    7E  7C 00096    CLRQ    -(SP)           ; 0293
    7E  D4 00098    CLRL    -(SP)           ;
00000000'  EF  9F 0009A    PUSHAB  STATUS          ;
    7E  7C 000A0    CLRQ    -(SP)           ;
    7E  D4 000A2    CLRL    -(SP)           ;
00000000'  EF  9F 000A4    PUSHAB  OUT             ;
    7E  D4 000AA    CLRL    -(SP)           ;
00000000'  EF  9F 000AC    PUSHAB  CMD             ;
00000000G  EF      0A  FB 000B2    CALLS   #10, LIB$SPAWN          ;
00000000'  EF      50  D0 000B9    MOVL    R0, IOSTATUS            ;
    1C 00000000'  EF  E8 000C0    BLBS    IOSTATUS, 4$          ; 0294
    40 0000FE02   9F      01  E1 000C7    BBC     #1, a#^X0000FE02, 6$          ;
00000000'  EF  DD 000CF    PUSHL   IOSTATUS            ;
00000000'  EF  9F 000D5    PUSHAB  P.AAG           ;
00000000'  EF  9F 000DB    PUSHAB  P.AAF           ;
    21  11 000E1    BRB     5$              ;
    52 00000000'  EF  E8 000E3 4$:   BLBS    STATUS, 7$            ; 0295
    1D 0000FE02   9F      01  E1 000EA    BBC     #1, a#^X0000FE02, 6$          ;
00000000'  EF  DD 000F2    PUSHL   STATUS          ;
00000000'  EF  9F 000F8    PUSHAB  P.AAI           ;
00000000'  EF  9F 000FE    PUSHAB  P.AAH           ;
    01  DD 00104 5$:   PUSHL   #1              ;
    1A  DD 00106    PUSHL   #26             ;
00000000G  FF      05  FB 00108    CALLS   #5, @CRD$PRINT          ;
00000000G  EF  9F 0010F 6$:   PUSHAB  MEN$GT_FAILED_ONL_AUTOSIZER       ;
    01  DD 00115    PUSHL   #1              ;
    1A  DD 00117    PUSHL   #26             ;
00000000G  FF      03  FB 00119    CALLS   #3, @CRD$PRINT          ;
```

```
        00000000G  EF  9F 00120     PUSHAB  CRD$GT_AUTOSIZER_ERROR        ;
            01  DD 00126     PUSHL   #1              ;
            1A  DD 00128     PUSHL   #26             ;
        00000000G  FF       03  FB 0012A     CALLS   #3, @CRD$PRINT          ;
            0E  DD 00131     PUSHL   #14             ;
        00000000G  EF       01  FB 00133     CALLS   #1, CRD$STOP            ;
            04  11 0013A     BRB     8$              ;
            50       01  D0 0013C 7$:   MOVL    #1, R0          ; 0298
             04 0013F     RET                     ;
            50  D4 00140 8$:   CLRL    R0              ; 0299
             04 00142     RET                     ;

; Routine Size:  323 bytes,    Routine Base:  CODE + 0017
```

```
;    0300  1 %SBTTL 'MM$Set_Flags_AutoSizer Routine'
;    0301  1
;    0302  1 GLOBAL ROUTINE MM$Set_Flags_AutoSizer (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0303  1
;    0304  1 !++
;    0305  1 ! Functional Description:
;    0306  1 !
;    0307  1 ! Describe offline function of routine
;    0308  1 !
;    0309  1 ! Online mode causes the file created in MM$LOAD_AUTOSIZER to be opened
;    0310  1 ! for reading in the MM$START_AUTOSIZER routine.
;    0311  1 !
;    0312  1 ! Formal Input Parameters:
;    0313  1 !
;    0314  1 ! None
;    0315  1 !
;    0316  1 ! Formal Output Parameters:
;    0317  1 !
;    0318  1 ! None
;    0319  1 !
;    0320  1 ! Side Effects:
;    0321  1 !
;    0322  1 ! None
;    0323  1 !
;    0324  1 ! Completion Codes:
;    0325  1 !
;    0326  1 ! CRDK$_Failure
;    0327  1 ! CRD$K_Success
;    0328  1 ! CRD$K_Repeat_Turing
;    0329  1 ! (Byte_count*1000h) + CRD$K_Success
;    0330  1 !--
```

```
;    0331  2 BEGIN     ! Routine MM$Set_Flags_AutoSizer
;    0332  2
;    0333  2 !**************************************************************************
;    0334  2 !+
;    0335  2 ! These definitions are for the RMS file operations. The file was produced
;    0336  2 ! by the ONLINE AUTOSIZER in the MM$LOAD_AUTOSIZER routine.
;    0337  2 !-
;    0338  2 !**************************************************************************
;    0339  2
;    0340  2 GLOBAL
;    0341  2     CRD$GA_Rec_Buf  : BLOCK [ EVSBB$S_EVSBB$REC, BYTE ],! RMS record buffer
; P  0342  2     CRD$GA_File_FAB : $FAB ( RFM=VAR,  ! Variable record length
; P  0343  2         RAT=CR,   ! Carriage control
;    0344  2         FAC=GET ),  ! Readonly access
;    0345  2     CRD$GA_File_RAB : $RAB ( USZ=EVSBB$S_EVSBB$REC ); ! Record size
;    0346  2
;    0347  2 !**************************************************************************
;    0348  2 !+
;    0349  2 ! Routine definitions
;    0350  2 !-
;    0351  2 !**************************************************************************
;    0352  2
;    0353  2 LITERAL
;    0354  2     FAO_BUFF_Size = 80;
;    0355  2 OWN
;    0356  2     Status : LONG,
;    0357  2     FAO_BUFF : BLOCK [FAO_BUFF_Size,BYTE],
;    0358  2     FAO_Length : WORD,
;    0359  2     FAO_Input_Dsc : BLOCK [2,LONG],
;    0360  2     FAO_Output_Dsc : BLOCK [2,LONG];
;    0361  2
;    0362  2 MAP
;    0363  2     CRD$GT_AutoSizer_Set_Flags : VECTOR [, BYTE],
;    0364  2     CRD$GT_DS_SET_FLAGS_Com    : VECTOR [, BYTE];
;    0365  2
;    0366  2 !**************************************************************************
;    0367  2 !+
;    0368  2 ! Program start
;    0369  2 ! IF (Menutest Offline mode) THEN
;    0370  2 ! Set the command line to the CLI to SET FLAGS ...
;    0371  2 !-
;    0372  2 !**************************************************************************
;    0373  2
;    0374  3 IF (.DSA$V_CRD_Menutest_Off)
;    0375  2 THEN
;    0376  3     BEGIN
;    0377  4     IF (.CRD$GT_AutoSizer_Set_Flags [0] EQL 0)
;    0378  3     THEN
;    0379  4         RETURN (CRD$K_Repeat_Turing)
;    0380  3     ELSE
;    0381  4         BEGIN
;    0382  4  CH$COPY (
;    0383  4      .CRD$GT_DS_SET_FLAGS_Com [0],
;    0384  4      CH$PTR (CRD$GT_DS_SET_FLAGS_Com [1]),
;    0385  4      .CRD$GT_AutoSizer_Set_Flags [0],
```

```
;    0386  4       CH$PTR (CRD$GT_AutoSizer_Set_Flags [1]),
;    0387  4       %C' ',
;    0388  4       .CLI_Buffer_Length,
;    0389  4       CH$PTR (.CLI_Buffer_Address)
;    0390  4       );
;    0391  4   CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0392  4   $CRD_Return_Length_Status (CRD$K_Success);
;    0393  3   END;
;    0394  2      END;
```

```
;    0395  2 !**************************************************************************
;    0396  2 !+
;    0397  2 ! IF (Menutest Online mode) THEN
;    0398  2 !     OPEN the Online Autoszier file for read operations to be used by
;    0399  2 !     MM$START_AUTOSIZER routine.
;    0400  2 ! -
;    0401  2 !**************************************************************************
;    0402  2
;    0403  3 IF (.DSA$V_CRD_Menutest_On)
;    0404  2 THEN
;    0405  3     BEGIN        ! [03]
;    0406  3     CRD$GA_File_FAB [ FAB$L_FNA ] = UPLIT ( %ASCIC 'EVSBB.DAT'); ! [03]
;    0407  3     CRD$GA_File_FAB [ FAB$B_FNS ] = ..CRD$GA_File_FAB[FAB$L_FNA]; ! [03]
;    0408  3     CRD$GA_File_FAB [ FAB$L_FNA ] = .CRD$GA_File_FAB[FAB$L_FNA]+1; ! [03]
;    0409  3
;    0410  3     CRD$GA_File_RAB [ RAB$L_FAB ] = CRD$GA_File_FAB;   ! [03]
;    0411  3     CRD$GA_File_RAB [ RAB$L_UBF ] = CRD$GA_Rec_Buf;    ! [03]
;    0412  3
;    0413  3     Status = $OPEN (FAB=CRD$GA_File_FAB );     ! [03]
;    0414  3     $Check_OA_status (Status);       ! [03]
;    0415  3
;    0416  3     Status = $CONNECT (RAB=CRD$GA_File_RAB);     ! [03]
;    0417  3     $Check_OA_status (Status);       ! [03]
;    0418  3
;    0419  3 !   Attach the correct CPU as the autosizer can't do this
;    0420  3 !   Attach with default commands, as we don't have the real data
;    0421  3
;    0422  3     SELECTONE .DSA$GL_SID<24,8,0> OF
;    0423  3  SET
;    0424  3  [ PR$_SID_TYP8SS ] : ! KA820
; P  0425  3   $LOAD_ASCID_S
; P  0426  3       (
; P  0427  3       NAM=FAO_Input_Dsc,
; P  0428  3       STR='ATTACH KA820 HUB KA0 YES YES 1FFF 1 0'
;    0429  3       );
;    0430  3  [ PR$_SID_TYP790 ] :  ! KA790
; P  0431  3   $LOAD_ASCID_S
; P  0432  3       (
; P  0433  3       NAM=FAO_Input_Dsc,
; P  0434  3       STR='ATTACH KA86 HUB KA0 YES YES 1FFF 1'
;    0435  3       );
;    0436  3  [ PR$_SID_TYP780 ] :  ! KA780
; P  0437  3   $LOAD_ASCID_S
; P  0438  3       (
; P  0439  3       NAM=FAO_Input_Dsc,
; P  0440  3       STR='ATTACH KA780 HUB KA0 YES YES 1FFF 1'
;    0441  3       );
;    0442  3  [ PR$_SID_TYP750 ] : ! KA750
; P  0443  3   $LOAD_ASCID_S
; P  0444  3       (
; P  0445  3       NAM=FAO_Input_Dsc,
; P  0446  3       STR='ATTACH KA750 HUB KA0 YES YES YES 1FFF 1'
;    0447  3       );
;    0448  3  [ PR$_SID_TYP730 ] : ! KA730
; P  0449  3   $LOAD_ASCID_S
```

```
;P 0450  3       (
;P 0451  3       NAM=FAO_Input_Dsc,
;P 0452  3       STR='ATTACH KA730 HUB KA0 YES 0 1 0 YES NO'
;   0453  3       );
;   0454  3  [ OTHERWISE      ] : !Don't attach anything
;   0455  3   RETURN CRD$K_Success
;   0456  3   TES;
;   0457  3
;   0458  3 !   Set up the output descriptor dynamically, as code must be PIC.
;   0459  3
;   0460  3     $LOAD_ASCID_A (NAM=FAO_Output_Dsc, SIZ=FAO_Buff_Size, ADR=FAO_Buff);
;   0461  3
;   0462  3 !   Process the string
;   0463  3
;   0464  3     Status = $FAO ( FAO_Input_Dsc, FAO_Length, FAO_Output_Dsc );
;   0465  3     IF (NOT .Status) THEN RETURN CRD$K_Success;
;   0466  3
;   0467  3 !   Move the string to the CLI buffer
;   0468  3
;   0469  3     CH$COPY   ( .FAO_Length, CH$PTR(FAO_BUFF),  %C' ',
;   0470  3    .CLI_Buffer_Length, CH$PTR(.CLI_Buffer_Address) );
;   0471  3     CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;   0472  3     $CRD_Return_Length_Status (CRD$K_Success);
;   0473  2     END;
;   0474  2 RETURN CRD$K_Success;
;   0475  1 END;     ! Routine MM$Set_Flags_AutoSizer


    .PSECT  WORK,NOEXE,2

    00018 CRD$GA_REC_BUF::
    .BLKB   115
    0008B   .BLKB   1
03  0008C CRD$GA_FILE_FAB::
    .BYTE   3           ;
50  0008D     .BYTE   80          ;
     0000  0008E     .WORD   0           ;
 00000000  00090     .LONG   0           ;
 00000000  00094     .LONG   0           ;
 00000000  00098     .LONG   0           ;
 00000000  0009C     .LONG   0           ;
     0000  000A0     .WORD   0           ;
02  000A2     .BYTE   2           ;
00  000A3     .BYTE   0           ;
 00000000  000A4     .LONG   0           ;
00  000A8     .BYTE   0           ;
00  000A9     .BYTE   0           ;
02  000AA     .BYTE   2           ;
02  000AB     .BYTE   2           ;
 00000000  000AC     .LONG   0           ;
 00000000  000B0     .LONG   0           ;
 00000000  000B4     .LONG   0           ;
 00000000  000B8     .LONG   0           ;
 00000000  000BC     .LONG   0           ;
```

```
        00  000C0     .BYTE   0           ;
        00  000C1     .BYTE   0           ;
             0000  000C2     .WORD   0           ;
         00000000  000C4     .LONG   0           ;
             0000  000C8     .WORD   0           ;
        00  000CA     .BYTE   0           ;
        00  000CB     .BYTE   0           ;
         00000000  000CC     .LONG   0           ;
         00000000  000D0     .LONG   0           ;
             0000  000D4     .WORD   0           ;
        00  000D6     .BYTE   0           ;
        00  000D7     .BYTE   0           ;
         00000000  000D8     .LONG   0           ;
        01  000DC CRD$GA_FILE_RAB::
            .BYTE   1           ;
        44  000DD     .BYTE   68          ;
             0000  000DE     .WORD   0           ;
         00000000  000E0     .LONG   0           ;
         00000000  000E4     .LONG   0           ;
         00000000  000E8     .LONG   0           ;
             0000# 000EC     .WORD   0[3]         ;
             0000  000F2     .WORD   0           ;
         00000000  000F4     .LONG   0           ;
             0000  000F8     .WORD   0           ;
        00  000FA     .BYTE   0           ;
        00  000FB     .BYTE   0           ;
             0073  000FC     .WORD   115         ;
             0000  000FE     .WORD   0           ;
         00000000  00100     .LONG   0           ;
         00000000  00104     .LONG   0           ;
         00000000  00108     .LONG   0           ;
         00000000  0010C     .LONG   0           ;
       00  00110     .BYTE   0           ;
       00  00111     .BYTE   0           ;
       00  00112     .BYTE   0           ;
       00  00113     .BYTE   0           ;
         00000000  00114     .LONG   0           ;
         00000000  00118     .LONG   0           ;
         00000000  0011C     .LONG   0           ;
            00120 STATUS: .BLKB   4
            00124 FAO_BUFF:
            .BLKB   80
            00174 FAO_LENGTH:
            .BLKB   2
            00176   .BLKB   2
            00178 FAO_INPUT_DSC:
            .BLKB   8
            00180 FAO_OUTPUT_DSC:
            .BLKB   8

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

     00  00  54  41  44  2E  42  42  53  56  45  09  00070 P.AAJ:  .ASCII  <9>\EVSBB.DAT\<0><0>        ;
41   21  20  2D  20  52  4F  52  52  45  20  2A  2A  2A  15  0007C P.AAK:  .ASCII  <21>\*** ERROR - !AC = !XL\       ;
     4C  58  21  20  3D  20  43  0008B             ;
```

```
                53  55  54  41  54  53  06  00092 P.AAL:  .ASCII  <6>\STATUS\          ;
41  21  20  2D  20  52  4F  52  52  45  20  2A  2A  2A  15  00099 P.AAM:  .ASCII  <21>\*** ERROR - !AC = !XL\       ;
                4C  58  21  20  3D  20  43  000A8                ;
                53  55  54  41  54  53  06  000AF P.AAN:  .ASCII  <6>\STATUS\          ;
                        000B6    .BLKB   2
55  48  20  30  32  38  41  4B  20  48  43  41  54  54  41  000B8 P.AAO:  .ASCII  \ATTACH KA820 HUB KA0 YES YES 1FFF 1 0\<0>  ;
31  20  53  45  59  20  53  45  59  20  30  41  4B  20  42  000C7           ;
                    00  30  20  31  20  46  46  46  000D6           ;
                        00  00  000DE    .ASCII  <0><0>          ;
42  55  48  20  36  38  41  4B  20  48  43  41  54  54  41  000E0 P.AAP:  .ASCII  \ATTACH KA86 HUB KA0 YES YES 1FFF 1\<0>     ;
46  31  20  53  45  59  20  53  45  59  20  30  41  4B  20  000EF           ;
                  00  31  20  46  46  000FE             ;
                    00  00103    .ASCII  <0>            ;
55  48  20  30  38  37  41  4B  20  48  43  41  54  54  41  00104 P.AAQ:  .ASCII  \ATTACH KA780 HUB KA0 YES YES 1FFF 1\<0>    ;
31  20  53  45  59  20  53  45  59  20  30  41  4B  20  42  00113           ;
                    00  31  20  46  46  46  00122             ;
55  48  20  30  35  37  41  4B  20  48  43  41  54  54  41  00128 P.AAR:  .ASCII  \ATTACH KA750 HUB KA0 YES YES YES 1FFF 1-   ;
59  20  53  45  59  20  53  45  59  20  30  41  4B  20  42  00137    \<0>          ;
                  00  31  20  46  46  46  31  20  53  45  00146          ;
55  48  20  30  33  37  41  4B  20  48  43  41  54  54  41  00150 P.AAS:  .ASCII  \ATTACH KA730 HUB KA0 YES 0 1 0 YES NO\<0>  ;
30  20  31  20  30  20  53  45  59  20  30  41  4B  20  42  0015F           ;
                    00  4F  4E  20  53  45  59  20  0016E           ;
                        00  00  00176    .ASCII  <0><0>         ;

                .EXTRN  CRD$GT_DS_SET_FLAGS_COM
                .EXTRN  SYS$OPEN, SYS$CONNECT
                .EXTRN  SYS$FAO

                .PSECT  CODE,NOWRT,  SHR,  PIC,2

                  03FC 00000    .ENTRY  MM$SET_FLAGS_AUTOSIZER, Save R2,R3,R4,R5,-  ; 0302
                 R6,R7,R8,R9            ;
                3E 0000FE02   9F  E9 00002    BLBC    a#^X0000FE02, 3$          ; 0374
                59 00000000'  EF  9A 00009    MOVZBL  CRD$GT_AUTOSIZER_SET_FLAGS, R9       ; 0377
                04  12 00010    BNEQ    1$              ;
                50      02  D0 00012    MOVL    #2, R0          ; 0381
            04 00015    RET             ;
                58 00000000G  EF  9A 00016 1$:   MOVZBL  CRD$GT_DS_SET_FLAGS_COM, R8        ; 0383
                57 04   AC  D0 0001D    MOVL    CLI_BUFFER_LENGTH, R7          ; 0388
                56 08   AC  D0 00021    MOVL    CLI_BUFFER_ADDRESS, R6         ; 0389
57              20 00000000G  EF      58  2C 00025    MOVC5   R8, CRD$GT_DS_SET_FLAGS_COM+1, #32, R7, -   ;
                66      0002E    (R6)            ;
                03  19 0002F    BLSS    2$              ;
             01D8  31 00031    BRW     14$             ;
                56      58  C0 00034 2$:   ADDL2   R8, R6          ;
                57      58  C2 00037    SUBL2   R8, R7          ;
57              20 00000000'  EF      59  2C 0003A    MOVC5   R9, CRD$GT_AUTOSIZER_SET_FLAGS+1, #32, R7, -;
                66      00043    (R6)            ;
             01C5  31 00044    BRW     14$             ; 0391
                03 0000FE02   9F      02  E0 00047 3$:   BBS     #2, a#^X0000FE02, 4$          ; 0403
             01CD  31 0004F    BRW     15$             ;
            00000000'  EF 00000000'  EF  9E 00052 4$:   MOVAB   P.AAJ, CRD$GA_FILE_FAB+44        ; 0406
            00000000'  EF 00000000'  FF  90 0005D    MOVB    aCRD$GA_FILE_FAB+44, CRD$GA_FILE_FAB+52     ; 0407
            00000000'  EF  D6 00068    INCL    CRD$GA_FILE_FAB+44          ; 0408
            00000000'  EF 00000000'  EF  9E 0006E    MOVAB   CRD$GA_FILE_FAB, CRD$GA_FILE_RAB+60       ; 0410
```

```
                00000000'  EF 00000000'  EF  9E 00079    MOVAB   CRD$GA_REC_BUF, CRD$GA_FILE_RAB+36      ; 0411
                00000000'  EF  9F 00084    PUSHAB  CRD$GA_FILE_FAB          ; 0413
                00000000G  00      01  FB 0008A    CALLS   #1, SYS$OPEN          ;
                00000000'  EF      50  D0 00091    MOVL    R0, STATUS           ;
                    1C 00000000'  EF  E8 00098    BLBS    STATUS, 5$           ; 0414
                    54 0000FE02   9F      01  E1 0009F    BBC     #1, @#^X0000FE02, 7$         ;
                00000000'  EF  DD 000A7    PUSHL   STATUS           ;
                00000000'  EF  9F 000AD    PUSHAB  P.AAL            ;
                00000000'  EF  9F 000B3    PUSHAB  P.AAK            ;
                    35  11 000B9    BRB     6$              ;
                00000000'  EF  9F 000BB 5$:   PUSHAB  CRD$GA_FILE_RAB          ; 0416
                00000000G  00      01  FB 000C1    CALLS   #1, SYS$CONNECT          ;
                00000000'  EF      50  D0 000C8    MOVL    R0, STATUS           ;
                    53 00000000'  EF  E8 000CF    BLBS    STATUS, 8$           ; 0417
                    1D 0000FE02   9F      01  E1 000D6    BBC     #1, @#^X0000FE02, 7$         ;
                00000000'  EF  DD 000DE    PUSHL   STATUS           ;
                00000000'  EF  9F 000E4    PUSHAB  P.AAN            ;
                00000000'  EF  9F 000EA    PUSHAB  P.AAM            ;
                    01  DD 000F0 6$:   PUSHL   #1              ;
                    1A  DD 000F2    PUSHL   #26             ;
                00000000G  FF      05  FB 000F4    CALLS   #5, @CRD$PRINT           ;
                00000000G  EF  9F 000FB 7$:   PUSHAB  MEN$GT_FAILED_ONL_AUTOSIZER        ;
                    01  DD 00101    PUSHL   #1              ;
                    1A  DD 00103    PUSHL   #26             ;
                00000000G  FF      03  FB 00105    CALLS   #3, @CRD$PRINT           ;
                00000000G  EF  9F 0010C    PUSHAB  CRD$GT_AUTOSIZER_ERROR       ;
                    01  DD 00112    PUSHL   #1              ;
                    1A  DD 00114    PUSHL   #26             ;
                00000000G  FF      03  FB 00116    CALLS   #3, @CRD$PRINT           ;
                    0E  DD 0011D    PUSHL   #14             ;
                00000000G  EF      01  FB 0011F    CALLS   #1, CRD$STOP             ;
                  00FA  31 00126    BRW     16$             ;
                    50 0000FE17   9F  9A 00129 8$:   MOVZBL  @#^X0000FE17, R0          ; 0422
                    05      50  91 00130    CMPB    R0, #5           ; 0424
                    18  12 00133    BNEQ    9$              ;
                00000000'  EF 010E0025   8F  D0 00135    MOVL    #17694757, FAO_INPUT_DSC        ; 0429
                00000000'  EF 00000000'  EF  9E 00140    MOVAB   P.AAO, FAO_INPUT_DSC+4       ;
                    72  11 0014B    BRB     13$             ; 0422
                    04      50  91 0014D 9$:   CMPB    R0, #4           ; 0430
                    18  12 00150    BNEQ    10$             ;
                00000000'  EF 010E0022   8F  D0 00152    MOVL    #17694754, FAO_INPUT_DSC        ; 0435
                00000000'  EF 00000000'  EF  9E 0015D    MOVAB   P.AAP, FAO_INPUT_DSC+4       ;
                    55  11 00168    BRB     13$             ; 0422
                    01      50  91 0016A 10$:   CMPB    R0, #1           ; 0436
                    18  12 0016D    BNEQ    11$             ;
                00000000'  EF 010E0023   8F  D0 0016F    MOVL    #17694755, FAO_INPUT_DSC        ; 0441
                00000000'  EF 00000000'  EF  9E 0017A    MOVAB   P.AAQ, FAO_INPUT_DSC+4       ;
                    38  11 00185    BRB     13$             ; 0422
                    02      50  91 00187 11$:   CMPB    R0, #2           ; 0442
                    18  12 0018A    BNEQ    12$             ;
                00000000'  EF 010E0027   8F  D0 0018C    MOVL    #17694759, FAO_INPUT_DSC        ; 0447
                00000000'  EF 00000000'  EF  9E 00197    MOVAB   P.AAR, FAO_INPUT_DSC+4       ;
                    1B  11 001A2    BRB     13$             ; 0422
                    03      50  91 001A4 12$:   CMPB    R0, #3           ; 0448
                    76  12 001A7    BNEQ    15$             ;
```

```
                00000000'  EF 010E0025   8F  D0 001A9     MOVL    #17694757, FAO_INPUT_DSC        ; 0453
                00000000'  EF 00000000'  EF  9E 001B4     MOVAB   P.AAS, FAO_INPUT_DSC+4          ;
                00000000'  EF 010E0050   8F  D0 001BF 13$:   MOVL    #17694800, FAO_OUTPUT_DSC       ; 0460
                00000000'  EF 00000000'  EF  9E 001CA     MOVAB   FAO_BUFF, FAO_OUTPUT_DSC+4      ;
                00000000'  EF  9F 001D5     PUSHAB  FAO_OUTPUT_DSC          ; 0464
                00000000'  EF  9F 001DB     PUSHAB  FAO_LENGTH              ;
                00000000'  EF  9F 001E1     PUSHAB  FAO_INPUT_DSC           ;
                00000000G  00      03  FB 001E7     CALLS   #3, SYS$FAO             ;
                00000000'  EF      50  D0 001EE     MOVL    R0, STATUS              ;
                    23 00000000'  EF  E9 001F5     BLBC    STATUS, 15$             ; 0465
            04  AC          20 00000000'  EF 00000000'  EF  2C 001FC     MOVC5   FAO_LENGTH, FAO_BUFF, #32, -    ; 0470
            08  BC      0020A     CLI_BUFFER_LENGTH, @CLI_BUFFER_ADDRESS      ;
                7E 04    AC  7D 0020C 14$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)        ; 0471
                00000000G  EF      02  FB 00210     CALLS   #2, CRD$PRINT_DS_COMMAND        ;
                50 00500001   8F  D0 00217     MOVL    #5242881, R0            ; 0472
                 04 0021E     RET                ;
                50       01  D0 0021F 15$:   MOVL    #1, R0          ; 0474
                 04 00222     RET                ;
                50  D4 00223 16$:   CLRL    R0              ; 0475
                 04 00225     RET                ;

; Routine Size:  550 bytes,    Routine Base:  CODE + 015A
```

```
;    0476  1
;    0477  1
;    0478  1 %SBTTL 'MM$Start_AutoSizer Routine'
;    0479  1
;    0480  1 GLOBAL ROUTINE MM$Start_AutoSizer (CLI_Buffer_Length, CLI_Buffer_Address) =
;    0481  1
;    0482  1 !++
;    0483  1 ! Functional Description:
;    0484  1 !
;    0485  1 ! Describe function of routine
;    0486  1 !
;    0487  1 ! Formal Input Parameters:
;    0488  1 !
;    0489  1 ! None
;    0490  1 !
;    0491  1 ! Formal Output Parameters:
;    0492  1 !
;    0493  1 ! None
;    0494  1 !
;    0495  1 ! Side Effects:
;    0496  1 !
;    0497  1 ! None
;    0498  1 !
;    0499  1 ! Completion Codes:
;    0500  1 !
;    0501  1 ! CRD$K_Success
;    0502  1 ! CRD$K_Failure
;    0503  1 ! (Byte_count*1000h) + CRD$K_Success
;    0504  1 !--
```

```
;    0505  2 BEGIN     ! Routine MM$Start_AutoSizer
;    0506  2
;    0507  2 !*************************************************************************
;    0508  2 !+
;    0509  2 ! Definitions
;    0510  2 !-
;    0511  2 !*************************************************************************
;    0512  2
;    0513  2 EXTERNAL CRD$GA_Rec_Buf  : ADDRESSING_MODE (LONG_RELATIVE);
;    0514  2 EXTERNAL CRD$GA_File_FAB : ADDRESSING_MODE (LONG_RELATIVE);
;    0515  2 EXTERNAL CRD$GA_File_RAB : ADDRESSING_MODE (LONG_RELATIVE);
;    0516  2
;    0517  2 MAP  CRD$GA_Rec_Buf  : BLOCK [,BYTE];
;    0518  2 MAP  CRD$GA_File_FAB : BLOCK [,BYTE];
;    0519  2 MAP  CRD$GA_File_RAB : BLOCK [,BYTE];
;    0520  2
;    0521  2 LITERAL
;    0522  2     Output_size = 80;
;    0523  2 OWN
;    0524  2     CNTRL : BLOCK [2,LONG],
;    0525  2     Output_desc : BLOCK [2,LONG] INITIAL (%X'010E0000'+Output_Size,0), ! [03]
;    0526  2     Output_Len : WORD,
;    0527  2     Status, Type : LONG,
;    0528  2     Output_Buff : VECTOR [Output_size,BYTE];
;    0529  2
;    0530  2 MAP
;    0531  2     CRD$GT_DS_START_Com : VECTOR [, BYTE];
;    0532  2
;    0533  2 !*************************************************************************
;    0534  2 !+
;    0535  2 ! Program start
;    0536  2 ! IF (Menutest Offline mode) THEN
;    0537  2 !   Load the VDS command buffer with SET FLAGS ... command.
;    0538  2 !-
;    0539  2 !*************************************************************************
;    0540  2
;    0541  3 IF (.DSA$V_CRD_Menutest_Off)
;    0542  2 THEN
;    0543  3     BEGIN
;    0544  3     CH$COPY (
;    0545  3  .CRD$GT_DS_START_Com [0],  CH$PTR (CRD$GT_DS_START_Com [1]),
;    0546  3  %C' ',
;    0547  3  .CLI_Buffer_Length,   CH$PTR (.CLI_Buffer_Address)
;    0548  3  );
;    0549  3     CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0550  3     $CRD Return_Length_Status (CRD$K_Success);
;    0551  2     END;
```

```
;    0552  2 !**************************************************************************
;    0553  2 !+
;    0554  2 ! IF (Menutest Online mode) THEN
;    0555  2 !   Read records from the AUTOSIZER file and create command line to
;    0556  2 !   be feed to VDS as ATTACH commands
;    0557  2 !-
;    0558  2 !**************************************************************************
;    0559  2
;    0560  3 IF (.DSA$V_CRD_Menutest_On)
;    0561  2 THEN
;    0562  3     BEGIN
;    0563  3 !   Read a record from the file opened in MM$Set_Flags_Autosizer
;    0564  3     Status = $GET(RAB=CRD$GA_File_RAB);
;    0565  3     SELECTONE .Status OF
;    0566  3  SET
;    0567  3  [ RMS$_EOF     ] : ;
;    0568  3 !    .    .   End of file condition, cleanup files by doing nothing
;    0569  3
;    0570  3  [ RMS$_NORMAL ] :
;    0571  4      BEGIN
;    0572  4 !    .    .   Get corresponding attach string
;    0573  4      Status = MEN$GENERATE_ONLINE_ATTACH
;    0574  4     ( .CLI_Buffer_Length, .CLI_Buffer_Address );
;    0575  4
;    0576  4 !    .    .   Fake out the MM state machine by backing up state till done
;    0577  4 !    .    .   reading the EVSBB.DAT file.
;    0578  4              CRD$GL_MM_Current_State = .CRD$GL_Previous_State;
;    0579  4
;    0580  4 !    .    .   IF (NO attach string) THEN just return all is well
;    0581  4      IF (.Status NEQ 0) THEN
;    0582  5   BEGIN
;    0583  5   CRD$Print_DS_Command (.CLI_Buffer_Length, .CLI_Buffer_Address);
;    0584  5       $CRD_Return_Length_Status (CRD$K_Success);
;    0585  5   END
;    0586  4      ELSE
;    0587  4          RETURN CRD$K_Success;
;    0588  3      END;
;    0589  3
;    0590  3  [ OTHERWISE ] :
;    0591  3 !    .    .   RMS file error
;    0592  3      $Check_OA_status (Status);     ! [03]
;    0593  3  TES;
;    0594  3
;    0595  3 !   Close t  file then delete it. This conserves disk space.
;    0596  3
;    0597  3     Status = $CLOSE (FAB=CRD$GA_File_FAB);
;    0598  3     $Check_OA_Status (Status);
;    0599  3     $ERASE (FAB=CRD$GA_File_FAB);
;    0600  3     $Check_OA_Status (Status);
;    0601  3     RETURN CRD$K_Success;  ! All has gone well
;    0602  2     END;
;    0603  2
;    0604  2 RETURN CRD$K_Failure   ! Should never get here
;    0605  1 END;     ! Routine MM$Start_AutoSizer
```

```
                .PSECT  WORK,NOEXE,2

                00188 CNTRL:    .BLKB   8
           00000000  010E0050  00190 OUTPUT_DESC:
                .LONG   17694800, 0             ;
                00198 OUTPUT_LEN:
                .BLKB   2
                0019A   .BLKB   2
                0019C STATUS:   .BLKB   4
                001A0 TYPE:     .BLKB   4
                001A4 OUTPUT_BUFF:
                .BLKB   80

                .PSECT  DATA,NOWRT,NOEXE,  SHR,2

41  21  20  2D  20  52  4F  52  52  45  20  2A  2A  2A  15  00178 P.AAT:   .ASCII  <21>\*** ERROR - !AC = !XL\          ;
    4C  58  21  20  3D  20  43  00187                   ;
    53  55  54  41  54  53  06  0018E P.AAU:   .ASCII  <6>\STATUS\                ;
41  21  20  2D  20  52  4F  52  52  45  20  2A  2A  2A  15  00195 P.AAV:   .ASCII  <21>\*** ERROR - !AC = !XL\          ;
    4C  58  21  20  3D  20  43  001A4                   ;
    53  55  54  41  54  53  06  001AB P.AAW:   .ASCII  <6>\STATUS\                ;
41  21  20  2D  20  52  4F  52  52  45  20  2A  2A  2A  15  001B2 P.AAX:   .ASCII  <21>\*** ERROR - !AC = !XL\          ;
    4C  58  21  20  3D  20  43  001C1                   ;
    53  55  54  41  54  53  06  001C8 P.AAY:   .ASCII  <6>\STATUS\                ;

                .EXTRN  CRD$GT_DS_START_COM
                .EXTRN  SYS$GET, CRD$GL_MM_CURRENT_STATE
                .EXTRN  CRD$GL_PREVIOUS_STATE
                .EXTRN  SYS$CLOSE, SYS$ERASE

                .PSECT  CODE,NOWRT,  SHR,  PIC,2

                  003C 00000     .ENTRY  MM$START_AUTOSIZER, Save R2,R3,R4,R5       ; 0480
                15 0000FE02   9F  E9 00002     BLBC    a#^X0000FE02, 1$            ; 0541
                50 00000000G  EF  9A 00009     MOVZBL  CRD$GT_DS_START_COM, R0      ; 0545
04  AC          20 00000000G  EF      50  2C 00010     MOVC5   R0, CRD$GT_DS_START_COM+1, #32, -       ; 0547
08  BC      0001A     CLI_BUFFER_LENGTH, @CLI_BUFFER_ADDRESS      ;
                60  11 0001C     BRB     3$              ; 0549
                03 0000FE02   9F      02  E0 0001E 1$:   BBS     #2, a#^X0000FE02, 2$        ; 0560
              0127  31 00026     BRW     11$             ;
             00000000G  EF  9F 00029 2$:   PUSHAB  CRD$GA_FILE_RAB          ; 0564
             00000000G  00      01  FB 0002F     CALLS   #1, SYS$GET             ;
             00000000'  EF      50  D0 00036     MOVL    R0, STATUS              ;
                52 00000000'  EF  D0 0003D     MOVL    STATUS, R2              ; 0565
             0001827A   8F      52  D1 00044     CMPL    R2, #98938              ; 0567
                62  13 0004B     BEQL    6$              ;
             00010001   8F      52  D1 0004D     CMPL    R2, #65537              ; 0570
                3B  12 00054     BNEQ    4$              ;
                7E 04   AC  7D 00056     MOVQ    CLI_BUFFER_LENGTH, -(SP)        ; 0574
             00000000G  EF      02  FB 0005A     CALLS   #2, MEN$GENERATE_ONLINE_ATTACH      ;
             00000000'  EF      50  D0 00061     MOVL    R0, STATUS              ;
             00000000G  EF 00000000G  EF  D0 00068     MOVL    CRD$GL_PREVIOUS_STATE, -        ; 0578
                 CRD$GL_MM_CURRENT_STATE        ;
```

```
                00000000'  EF  D5 00073    TSTL    STATUS                  ; 0581
                    03  12 00079    BNEQ    3$                  ;
                  00CE  31 0007B    BRW     10$                 ;
                    7E 04   AC  7D 0007E 3$:   MOVQ    CLI_BUFFER_LENGTH, -(SP)        ; 0583
                00000000G  EF       02  FB 00082    CALLS   #2, CRD$PRINT_DS_COMMAND        ;
                    50 00500001   8F  D0 00089    MOVL    #5242881, R0            ; 0587
                 04 00090    RET                 ;
                    1B       52  E8 00091 4$:   BLBS    R2, 6$              ; 0592
                    03 0000FE02   9F       01  E0 00094    BBS     #1, a#^X0000FE02, 5$        ;
                  0080  31 0009C    BRW     9$                  ;
                    52  DD 0009F 5$:   PUSHL   R2                  ;
                00000000'  EF  9F 000A1    PUSHAB  P.AAU               ;
                00000000'  EF  9F 000A7    PUSHAB  P.AAT               ;
                    65  11 000AD    BRB     8$                  ;
                00000000G  EF  9F 000AF 6$:   PUSHAB  CRD$GA_FILE_FAB         ; 0597
                00000000G  00       01  FB 000B5    CALLS   #1, SYS$CLOSE           ;
                00000000'  EF       50  D0 000BC    MOVL    R0, STATUS              ;
                    1C 00000000'  EF  E8 000C3    BLBS    STATUS, 7$              ; 0598
                    4D 0000FE02   9F       01  E1 000CA    BBC     #1, a#^X0000FE02, 9$        ;
                00000000'  EF  DD 000D2    PUSHL   STATUS              ;
                00000000'  EF  9F 000D8    PUSHAB  P.AAW               ;
                00000000'  EF  9F 000DE    PUSHAB  P.AAV               ;
                    2E  11 000E4    BRB     8$                  ;
                00000000G  EF  9F 000E6 7$:   PUSHAB  CRD$GA_FILE_FAB         ; 0599
                00000000G  00       01  FB 000EC    CALLS   #1, SYS$ERASE           ;
                    52 00000000'  EF  E8 000F3    BLBS    STATUS, 10$             ; 0600
                    1D 0000FE02   9F       01  E1 000FA    BBC     #1, a#^X0000FE02, 9$        ;
                00000000'  EF  DD 00102    PUSHL   STATUS              ;
                00000000'  EF  9F 00108    PUSHAB  P.AAY               ;
                00000000'  EF  9F 0010E    PUSHAB  P.AAX               ;
                    01  DD 00114 8$:   PUSHL   #1                  ;
                    1A  DD 00116    PUSHL   #26                 ;
                00000000G  FF       05  FB 00118    CALLS   #5, aCRD$PRINT          ;
                00000000G  EF  9F 0011F 9$:   PUSHAB  MEN$GT_FAILED_ONL_AUTOSIZER       ;
                    01  DD 00125    PUSHL   #1                  ;
                    1A  DD 00127    PUSHL   #26                 ;
                00000000G  FF       03  FB 00129    CALLS   #3, aCRD$PRINT          ;
                00000000G  EF  9F 00130    PUSHAB  CRD$GT_AUTOSIZER_ERROR        ;
                    01  DD 00136    PUSHL   #1                  ;
                    1A  DD 00138    PUSHL   #26                 ;
                00000000G  FF       03  FB 0013A    CALLS   #3, aCRD$PRINT          ;
                    0E  DD 00141    PUSHL   #14                 ;
                00000000G  EF       01  FB 00143    CALLS   #1, CRD$STOP            ;
                    04  11 0014A    BRB     11$                 ;
                    50       01  D0 0014C 10$:   MOVL    #1, R0              ; 0601
                 04 0014F    RET                 ;
                    50  D4 00150 11$:   CLRL    R0                  ; 0605
                 04 00152    RET                 ;

; Routine Size:  339 bytes,    Routine Base:  CODE + 0380
```

```
;    0606  1 %SBTTL 'MM$Build_DDBs Routine'
;    0607  1
;    0608  1 GLOBAL ROUTINE MM$Build_DDBs =
;    0609  1
;    0610  1 !++
;    0611  1 ! Functional Description:
;    0612  1 !
;    0613  1 ! Describe function of routine
;    0614  1 !
;    0615  1 ! Formal Input Parameters:
;    0616  1 !
;    0617  1 ! None
;    0618  1 !
;    0619  1 ! Formal Output Parameters:
;    0620  1 !
;    0621  1 ! None
;    0622  1 !
;    0623  1 ! Side Effects:
;    0624  1 !
;    0625  1 ! None
;    0626  1 !
;    0627  1 ! Completion Codes:
;    0628  1 !
;    0629  1 ! None
;    0630  1 !--
```

```
;    0631  2 BEGIN     ! Routine MM$Build_DDBs
;    0632  2  $CRD_Print (MEN$GT_AutoSizer_End);
;    0633  2       ! Tell the operator is AutoSizer is done.
;    0634  2
;    0635  2  CRD$Screen_Pause (SCREEN$K_Press_Return);        ! [02]
;    0636  2       ! Tell the soft console operator to
;    0637  2       ! Press Return for More...
;    0638  2       ! Before continuing with Menu mode.
;    0639  2
;    0640  2  MEN$Build_DDB ();
;    0641  2       ! Build the Device Data Blocks
;    0642  2
;    0643  3  RETURN (CRD$K_Repeat_Turing)
;    0644  1 END;     ! Routine MM$Build_DDBs



          .EXTRN  MEN$GT_AUTOSIZER_END
          .EXTRN  CRD$SCREEN_PAUSE
          .EXTRN  MEN$BUILD_DDB

                0000 00000     .ENTRY  MM$BUILD_DDBS, Save nothing        ; 0608
            00000000G  EF  9F 00002     PUSHAB  MEN$GT_AUTOSIZER_END        ; 0632
                01  DD 00008     PUSHL   #1                 ;
                1A  DD 0000A     PUSHL   #26                ;
            00000000G  FF       03  FB 0000C     CALLS   #3, @CRD$PRINT          ;
                7E  D4 00013     CLRL    -(SP)              ; 0635
            00000000G  EF       01  FB 00015     CALLS   #1, CRD$SCREEN_PAUSE        ;
            00000000G  EF       00  FB 0001C     CALLS   #0, MEN$BUILD_DDB         ; 0640
                50       02  D0 00023     MOVL    #2, R0             ; 0643
             04 00026     RET                 ; 0644

; Routine Size:  39 bytes,    Routine Base:  CODE + 04D3
```

```
;    0645  1 %SBTTL 'MM$Build_HSTs Routine'
;    0646  1
;    0647  1 GLOBAL ROUTINE MM$Build_HSTs =
;    0648  1
;    0649  1 !++
;    0650  1 ! Functional Description:
;    0651  1 !
;    0652  1 ! Describe function of routine
;    0653  1 !
;    0654  1 ! Formal Input Parameters:
;    0655  1 !
;    0656  1 ! None
;    0657  1 !
;    0658  1 ! Formal Output Parameters:
;    0659  1 !
;    0660  1 ! None
;    0661  1 !
;    0662  1 ! Side Effects:
;    0663  1 !
;    0664  1 ! None
;    0665  1 !
;    0666  1 ! Completion Codes:
;    0667  1 !
;    0668  1 ! None
;    0669  1 ! -
```

```
;   0670  2 BEGIN     ! Routine MM$Build_HSTs
;   0671  2  MEN$Determine_Support ();
;   0672  2  MEN$Specify_SelectNo ();
;   0673  3  RETURN (CRD$K_Repeat_Turing)
;   0674  1 END;      ! Routine MM$Build_HSTs



            .EXTRN  MEN$DETERMINE_SUPPORT
            .EXTRN  MEN$SPECIFY_SELECTNO

              0000 00000     .ENTRY  MM$BUILD_HSTS, Save nothing        ; 0647
        00000000G  EF      00  FB 00002    CALLS   #0, MEN$DETERMINE_SUPPORT        ; 0671
        00000000G  EF      00  FB 00009    CALLS   #0, MEN$SPECIFY_SELECTNO         ; 0672
           50       02  D0 00010    MOVL    #2, R0              ; 0673
         04 00013    RET                 ; 0674

; Routine Size:  20 bytes,    Routine Base:  CODE + 04FA
```

```
;    0675  1 %SBTTL 'MM$Done_Inits Routine'
;    0676  1
;    0677  1 GLOBAL ROUTINE MM$Done_Inits =
;    0678  1
;    0679  1 !++
;    0680  1 ! Functional Description:
;    0681  1 !
;    0682  1 ! Describe function of routine
;    0683  1 !
;    0684  1 ! Formal Input Parameters:
;    0685  1 !
;    0686  1 ! None
;    0687  1 !
;    0688  1 ! Formal Output Parameters:
;    0689  1 !
;    0690  1 ! None
;    0691  1 !
;    0692  1 ! Side Effects:
;    0693  1 !
;    0694  1 ! None
;    0695  1 !
;    0696  1 ! Completion Codes:
;    0697  1 !
;    0698  1 ! None
;    0699  1 !--
```

```
;   0700  2 BEGIN     ! Routine MM$Done_Inits
;   0701  3  RETURN (CRD$K_Repeat_Turing)
;   0702  1 END;      ! Routine MM$Done_Inits



                0000 00000     .ENTRY  MM$DONE_INITS, Save nothing        ; 0677
            50      02  D0 00002     MOVL    #2, R0              ; 0701
        04 00005     RET                 ; 0702

; Routine Size:  6 bytes,    Routine Base:  CODE + 050E
```

```
;    0703  1 %SBTTL 'MM$Print_Menu Routine'
;    0704  1
;    0705  1 GLOBAL ROUTINE MM$Print_Menu =
;    0706  1
;    0707  1 !++
;    0708  1 ! Functional Description:
;    0709  1 !
;    0710  1 ! Describe function of routine
;    0711  1 !
;    0712  1 ! Formal Input Parameters:
;    0713  1 !
;    0714  1 ! None
;    0715  1 !
;    0716  1 ! Formal Output Parameters:
;    0717  1 !
;    0718  1 ! None
;    0719  1 !
;    0720  1 ! Side Effects:
;    0721  1 !
;    0722  1 ! None
;    0723  1 !
;    0724  1 ! Completion Codes:
;    0725  1 !
;    0726  1 ! None
;    0727  1 !--
```

```
;   0728  2 BEGIN     ! Routine MM$Print_Menu
;   0729  2  DSA$V_Oper = True;  ! Do this to indicate that an Operator is present to answer the prompts.
;   0730  2  MEN$FuncTest_Menu ();  ! Call the Menu routine
;   0731  3  RETURN (CRD$K_Repeat_Turing) ! Repeat the Turing_Machine routine
;   0732  1 END;     ! Routine MM$Print_Menu



          .EXTRN  MEN$FUNCTEST_MENU

            0000 00000     .ENTRY  MM$PRINT_MENU, Save nothing        ; 0705
       0000FE01   9F       10  88 00002    BISB2   #16, a#^X0000FE01          ; 0729
       00000000G  EF       00  FB 00009    CALLS   #0, MEN$FUNCTEST_MENU          ; 0730
           50      02  D0 00010    MOVL    #2, R0           ; 0731
        04 00013    RET                 ; 0732

; Routine Size:  20 bytes,    Routine Base:  CODE + 0514
```

```
;    0733  1 %SBTTL 'MM$Change_Table Routine'
;    0734  1
;    0735  1 GLOBAL ROUTINE MM$Change_Table =
;    0736  1
;    0737  1 !++
;    0738  1 ! Functional Description:
;    0739  1 !
;    0740  1 ! Describe function of routine
;    0741  1 !
;    0742  1 ! Formal Input Parameters:
;    0743  1 !
;    0744  1 ! None
;    0745  1 !
;    0746  1 ! Formal Output Parameters:
;    0747  1 !
;    0748  1 ! None
;    0749  1 !
;    0750  1 ! Side Effects:
;    0751  1 !
;    0752  1 ! None
;    0753  1 !
;    0754  1 ! Completion Codes:
;    0755  1 !
;    0756  1 ! None
;    0757  1 !--
```

```
;    0758  2 BEGIN     ! Routine MM$Change_Table
;    0759  2  CRD$GB_Current_State_Table = MEN$K_TQ_State_Table;
;    0760  2       ! Before using this, always MAP REF it to the appropriate structure.
;    0761  3  RETURN (CRD$K_Repeat_Turing)
;    0762  1 END;     ! Routine MM$Change_Table



            .EXTRN  CRD$GB_CURRENT_STATE_TABLE

              0000 00000     .ENTRY  MM$CHANGE_TABLE, Save nothing          ; 0735
        00000000G  EF       01  90 00002     MOVB    #1, CRD$GB_CURRENT_STATE_TABLE        ; 0759
            50       02  D0 00009     MOVL    #2, R0              ; 0761
           04 0000C     RET                 ; 0762

; Routine Size:  13 bytes,    Routine Base:  CODE + 0528
```

```
;    0763  1 %SBTTL 'MM$Menu_Prompt Routine'
;    0764  1
;    0765  1 GLOBAL ROUTINE MM$Menu_Prompt =
;    0766  1
;    0767  1 !++
;    0768  1 ! Functional Description:
;    0769  1 !
;    0770  1 ! Describe function of routine
;    0771  1 !
;    0772  1 ! Formal Input Parameters:
;    0773  1 !
;    0774  1 ! None
;    0775  1 !
;    0776  1 ! Formal Output Parameters:
;    0777  1 !
;    0778  1 ! None
;    0779  1 !
;    0780  1 ! Side Effects:
;    0781  1 !
;    0782  1 ! None
;    0783  1 !
;    0784  1 ! Completion Codes:
;    0785  1 !
;    0786  1 ! None
;    0787  1 !--
```

```
;    0788  2 BEGIN     ! Routine MM$Menu_Prompt
;    0789  2  !+
;    0790  2  ! Return failure so that the Main Menu prompt will go ahead and prompt the user
;    0791  2  ! for a menu selection, rather than calling CRD again. That is, returning failure
;    0792  2  ! will cause the QIO to be executed, prompting the user for a menu selection.
;    0793  2  !-
;    0794  2
;    0795  3  RETURN (CRD$K_Return_Failure)
;    0796  1 END;     ! Routine MM$Menu_Prompt



              0000 00000     .ENTRY  MM$MENU_PROMPT, Save nothing        ; 0765
           50  D4 00002     CLRL    R0              ; 0795
        04 00004     RET                 ; 0796

; Routine Size:  5 bytes,    Routine Base:  CODE + 0535
```

```
;    0797  1 %SBTTL 'MM$Internal_Error Routine'
;    0798  1
;    0799  1 GLOBAL ROUTINE MM$Internal_Error (TypeCode, Length, Address) =
;    0800  1
;    0801  1 !++
;    0802  1 ! Functional Description:
;    0803  1 !
;    0804  1 ! Describe function of routine
;    0805  1 !
;    0806  1 ! Formal Input Parameters:
;    0807  1 !
;    0808  1 ! None
;    0809  1 !
;    0810  1 ! Formal Output Parameters:
;    0811  1 !
;    0812  1 ! None
;    0813  1 !
;    0814  1 ! Side Effects:
;    0815  1 !
;    0816  1 ! None
;    0817  1 !
;    0818  1 ! Completion Codes:
;    0819  1 !
;    0820  1 ! None
;    0821  1 !--
```

```
;   0822  2 BEGIN       ! Routine MM$Internal_Error
;   0823  2  MEN$Menu_Test_Internal_Error (CRD$K_MM_Internal_Error, .TypeCode, .Length, .Address);
;   0824  2  CRD$Stop (MENU$K_Exit_Internal_Error);
;   0825  3  RETURN (CRD$K_Return_Success)
;   0826  1 END;      ! Routine MM$Internal_Error



                0000 00000     .ENTRY  MM$INTERNAL_ERROR, Save nothing        ; 0799
            7E 08    AC  7D 00002     MOVQ    LENGTH, -(SP)                  ; 0823
        04  AC     DD 00006     PUSHL   TYPECODE                ;
            7E         06  CE 00009     MNEGL   #6, -(SP)               ;
         00000000V  EF         04  FB 0000C     CALLS   #4, MEN$MENU_TEST_INTERNAL_ERROR       ;
            0D   DD 00013     PUSHL   #13                    ; 0824
         00000000G  EF         01  FB 00015     CALLS   #1, CRD$STOP            ;
            50         01  D0 0001C     MOVL    #1, R0                 ; 0825
          04 0001F     RET                     ; 0826

; Routine Size:  32 bytes,    Routine Base:  CODE + 053A
```

```
;    0827  1 %SBTTL 'MM$AutoSizer_Error Routine'
;    0828  1
;    0829  1 GLOBAL ROUTINE MM$AutoSizer_Error =
;    0830  1
;    0831  1 !++
;    0832  1 ! Functional Description:
;    0833  1 !
;    0834  1 ! Describe function of routine
;    0835  1 !
;    0836  1 ! Formal Input Parameters:
;    0837  1 !
;    0838  1 ! None
;    0839  1 !
;    0840  1 ! Formal Output Parameters:
;    0841  1 !
;    0842  1 ! None
;    0843  1 !
;    0844  1 ! Side Effects:
;    0845  1 !
;    0846  1 ! None
;    0847  1 !
;    0848  1 ! Completion Codes:
;    0849  1 !
;    0850  1 ! None
;    0851  1 !--
```

```
;    0852  2 BEGIN     ! Routine MM$AutoSizer_Error
;    0853  2  CRD$AutoSizer_Error ();
;    0854  2  CRD$Stop (MENU$K_Exit_AutoSizer_Error);
;    0855  3  RETURN (CRD$K_Return_Failure)
;    0856  1 END;     ! Routine MM$AutoSizer_Error


            .EXTRN  CRD$AUTOSIZER_ERROR

                  0000 00000     .ENTRY  MM$AUTOSIZER_ERROR, Save nothing        ; 0829
            00000000G  EF        00  FB 00002     CALLS   #0, CRD$AUTOSIZER_ERROR         ; 0853
                0E  DD 00009     PUSHL   #14                ; 0854
            00000000G  EF        01  FB 0000B     CALLS   #1, CRD$STOP            ;
                50  D4 00012     CLRL    R0                ; 0855
                 04 00014     RET                     ; 0856

; Routine Size:  21 bytes,    Routine Base:  CODE + 055A
```

```
;    0857  1 %SBTTL 'MEN$Menu_Test_Internal_Error Routine'
;    0858  1
;    0859  1 GLOBAL ROUTINE MEN$Menu_Test_Internal_Error (Error_Type, TypeCode, Length, Address) : NOVALUE =
;    0860  1
;    0861  1 !++
;    0862  1 ! Functional Description:
;    0863  1 !
;    0864  1 ! Describe function of routine
;    0865  1 !
;    0866  1 ! Formal Input Parameters:
;    0867  1 !
;    0868  1 ! None
;    0869  1 !
;    0870  1 ! Formal Output Parameters:
;    0871  1 !
;    0872  1 ! None
;    0873  1 !
;    0874  1 ! Side Effects:
;    0875  1 !
;    0876  1 ! None
;    0877  1 !
;    0878  1 ! Completion Codes:
;    0879  1 !
;    0880  1 ! None
;    0881  1 !--
```

```
;    0882  2 BEGIN     ! Routine MEN$Menu_Test_Internal_Error
;    0883  2  MAP
;    0884  2   CRD$AL_CRD_Dispatch_Vectors : VECTOR [4, BYTE];
;    0885  2
;    0886  2  !+
;    0887  2  ! Check that these constants are okay.          ![01]
;    0888  2  !-
;    0889  2
; P  0890  2  CRD_Assert ((CRD$K_Total_Numb_Vectors EQL (CRD$K_Numb_UnUsed_Vectors + CRD$K_Numb_Dispatch_Vectors)), ![01]
; L  0891  2   'Check the CRD.R32 library where it defines these constants');     ![01]
; %PRINT: ASSERT: The assertion is true.
;    0892  2
;    0893  2  !+
;    0894  2  ! Increment the negative version number in the dispatch vector area.     ![01]
;    0895  2  !-
;    0896  2
;    0897  2  CRD$AL_CRD_Dispatch_Vectors [1] = .CRD$AL_CRD_Dispatch_Vectors [1] + 1;
;    0898  2
;    0899  2  !+
;    0900  2  ! Now store all the debugging information we can think of!       ![01]
;    0901  2  !-
;    0902  2
;    0903  2  CRD$GL_CRD_Debug_Vector [0] = CRD$K_Total_Numb_Vectors;  ! Store how many there are.  ![01]
;    0904  2
;    0905  2  CRD$GL_CRD_Debug_Vector [1] = .Error_Type;        ![01]
;    0906  2
;    0907  2  CRD$GL_CRD_Debug_Vector [2] = .TypeCode;         ![01]
;    0908  2  CRD$GL_CRD_Debug_Vector [3] = .Length;          ![01]
;    0909  2  CRD$GL_CRD_Debug_Vector [4] = .Address;          ![01]
;    0910  2
;    0911  2  CRD$GL_CRD_Debug_Vector [5] = .CRD$GB_Current_State_Table;      ![01]
;    0912  2  CRD$GL_CRD_Debug_Vector [6] = .CRD$GL_MM_Current_State;       ![01]
;    0913  2  CRD$GL_CRD_Debug_Vector [7] = .CRD$GL_TQ_Current_State;       ![01]
;    0914  2  CRD$GL_CRD_Debug_Vector [8] = .CRD$GL_HS_Current_State;       ![01]
;    0915  2  CRD$GL_CRD_Debug_Vector [9] = .CRD$GL_Previous_State;      ![01]
;    0916  2
;    0917  2  CRD$GL_CRD_Debug_Vector [10] = .DSA$GL_Flags;   ! DSA flags    ![01]
;    0918  2  CRD$GL_CRD_Debug_Vector [11] = ..CRD$GA_DS_Flags;  ! DS Flags    ![01]
;    0919  2
;    0920  2  CRD$GL_CRD_Debug_Vector [12] = .CRD$GA_HS_Current_DDB_Ptr;      ![01]
;    0921  2  CRD$GL_CRD_Debug_Vector [13] = .CRD$GA_HS_Current_HSB_Ptr;      ![01]
;    0922  2  CRD$GL_CRD_Debug_Vector [14] = .CRD$GL_HS_Current_HSB_Numb;      ![01]
;    0923  2
;    0924  2  CRD$GL_CRD_Debug_Vector [15] = .CRD$GL_TQ_Current_TQ_Entry;      ![01]
;    0925  2
;    0926  2  CRD$GL_CRD_Debug_Vector [16] = CRD$AL_CRD_Dispatch_Vectors; ! Starting address of CRD  ![01]
;    0927  2
;    0928  2  !+
;    0929  2  ! Now, in the rest of the debug vectors, store the program counters of the last 'n' call frames. Note ![01]
;    0930  2  ! 'n' is dependent on how much information is stored above (i.e., how many debug vectors are left at ![01]
;    0931  2  ! this point).            ![01]
;    0932  2  !-
;    0933  2
;    0934  3  BEGIN              ![01]
;    0935  3   BUILTIN            ![01]
```

```
;    0936  3     PROBEW,       ! Probe write-accessability   ![01]
;    0937  3     PC,       ! Program Counter    ![01]
;    0938  3     FP;       ! Frame pointer    ![01]
;    0939  3
;    0940  3   LOCAL               ![01]
;    0941  3     Program_Counter,      ! ...      ![01]
;    0942  3     Frame_Ptr;      ! Temp frame pointer    ![01]
;    0943  3
;    0944  3   INCR Index FROM 17 TO (CRD$K_Total_Numb_Vectors - 1) DO      ![01]
;    0945  3     CRD$GL_CRD_Debug_Vector [.Index] = 0;  ! Default to 0     ![01]
;    0946  3
;    0947  3   Frame_Ptr = .FP;      ! Initialize to current frame   ![01]
;    0948  3   Program_Counter = MEN$Menu_Test_Internal_Error;  ! Initialize to this rouitne  ![01]
;    0949  3 !   Program_Counter = .PC;      ! Initialize to here    ![01]
;    0950  3
;    0951  3   INCR Index FROM 17 TO (CRD$K_Total_Numb_Vectors - 1) DO      ![01]
;    0952  4     BEGIN               ![01]
;    0953  4     CRD$GL_CRD_Debug_Vector [.Index] = .Program_Counter;      ![01]
;    0954  4           ! Store this PC in the debug vector  ![01]
;    0955  4
;    0956  4     Frame_Ptr = .(.Frame_Ptr + 12);    ! Get next frame pointer   ![01]
;    0957  4
;    0958  4     IF (.Frame_Ptr EQL 0) THEN    ! If at the end of the linked list of frames,  ![01]
;    0959  4      EXITLOOP;     ! ... exit this loop    ![01]
;    0960  4
;    0961  4     IF (NOT PROBEW (%REF (0), %REF (3), .Frame_Ptr)) THEN      ![01]
;    0962  4      EXITLOOP;     ! If we can't read it, exit this loop  ![01]
;    0963  4
;    0964  4     Program_Counter = .(.Frame_Ptr + 16);  ! Pick out PC from next frame   ![01]
;    0965  3     END;              ![01]
;    0966  2   END;                ![01]
;    0967  1 END;      ! Routine MEN$Menu_Test_Internal_Error


        .EXTRN  CRD$AL_CRD_DISPATCH_VECTORS
        .EXTRN  CRD$GL_CRD_DEBUG_VECTOR
        .EXTRN  CRD$GL_TQ_CURRENT_STATE
        .EXTRN  CRD$GL_HS_CURRENT_STATE
        .EXTRN  CRD$GA_DS_FLAGS
        .EXTRN  CRD$GA_HS_CURRENT_DDB_PTR
        .EXTRN  CRD$GA_HS_CURRENT_HSB_PTR
        .EXTRN  CRD$GL_HS_CURRENT_HSB_NUMB
        .EXTRN  CRD$GL_TQ_CURRENT_TQ_ENTRY

          0004 00000     .ENTRY  MEN$MENU_TEST_INTERNAL_ERROR, Save R2       ; 0859
00000000G   EF  96 00002     INCB    CRD$AL_CRD_DISPATCH_VECTORS+1          ; 0897
00000000G   EF       19  D0 00008     MOVL    #25, CRD$GL_CRD_DEBUG_VECTOR        ; 0903
00000000G   EF 04    AC  7D 0000F     MOVQ    ERROR_TYPE, CRD$GL_CRD_DEBUG_VECTOR+4       ; 0905
00000000G   EF 0C    AC  7D 00017     MOVQ    LENGTH, CRD$GL_CRD_DEBUG_VECTOR+12      ; 0908
00000000G   EF 00000000G   EF  9A 0001F     MOVZBL  CRD$GB_CURRENT_STATE_TABLE, -       ; 0911
        CRD$GL_CRD_DEBUG_VECTOR+20          ;
00000000G   EF 00000000G   EF  D0 0002A     MOVL    CRD$GL_MM_CURRENT_STATE, -      ; 0912
        CRD$GL_CRD_DEBUG_VECTOR+24          ;
00000000G   EF 00000000G   EF  D0 00035     MOVL    CRD$GL_TQ_CURRENT_STATE, -      ; 0913
```

```
                CRD$GL_CRD_DEBUG_VECTOR+28      ;
00000000G  EF 00000000G  EF  D0 00040    MOVL    CRD$GL_HS_CURRENT_STATE, -      ; 0914
                CRD$GL_CRD_DEBUG_VECTOR+32      ;
00000000G  EF 00000000G  EF  D0 0004B    MOVL    CRD$GL_PREVIOUS_STATE, -        ; 0915
                CRD$GL_CRD_DEBUG_VECTOR+36      ;
00000000G  EF 0000FE00   9F  D0 00056    MOVL    @#^X0000FE00, CRD$GL_CRD_DEBUG_VECTOR+40    ; 0917
00000000G  EF 00000000G  FF  D0 00061    MOVL    @CRD$GA_DS_FLAGS, -             ; 0918
                CRD$GL_CRD_DEBUG_VECTOR+44      ;
00000000G  EF 00000000G  EF  D0 0006C    MOVL    CRD$GA_HS_CURRENT_DDB_PTR, -    ; 0920
                CRD$GL_CRD_DEBUG_VECTOR+48      ;
00000000G  EF 00000000G  EF  D0 00077    MOVL    CRD$GA_HS_CURRENT_HSB_PTR, -    ; 0921
                CRD$GL_CRD_DEBUG_VECTOR+52      ;
00000000G  EF 00000000G  EF  D0 00082    MOVL    CRD$GL_HS_CURRENT_HSB_NUMB, -   ; 0922
                CRD$GL_CRD_DEBUG_VECTOR+56      ;
00000000G  EF 00000000G  EF  D0 0008D    MOVL    CRD$GL_TQ_CURRENT_TQ_ENTRY, -   ; 0924
                CRD$GL_CRD_DEBUG_VECTOR+60      ;
00000000G  EF 00000000G  EF  9E 00098    MOVAB   CRD$AL_CRD_DISPATCH_VECTORS, -  ; 0926
                CRD$GL_CRD_DEBUG_VECTOR+64      ;
     50       11  D0 000A3    MOVL    #17, INDEX              ; 0944
00000000GEF40  D4 000A6 1$:   CLRL    CRD$GL_CRD_DEBUG_VECTOR[INDEX]          ; 0945
     F5         50       18  F3 000AD    AOBLEQ  #24, INDEX, 1$          ;
     51       5D  D0 000B1    MOVL    FP, FRAME_PTR           ; 0947
     52     FF48   CF  9E 000B4    MOVAB   MEN$MENU_TEST_INTERNAL_ERROR, -         ; 0948
      PROGRAM_COUNTER           ;
     50       11  D0 000B9    MOVL    #17, INDEX              ; 0951
00000000GEF40       52  D0 000BC 2$:   MOVL    PROGRAM_COUNTER, CRD$GL_CRD_DEBUG_VECTOR-    ; 0953
     [INDEX]            ;
     51 0C    A1  D0 000C4    MOVL    12(FRAME_PTR), FRAME_PTR        ; 0956
     0E   13 000C8    BEQL    3$              ; 0958
     61          03       00  0D 000CA    PROBEW  #0, #3, (FRAME_PTR)         ; 0961
     08   13 000CE    BEQL    3$              ;
     52 10    A1  D0 000D0    MOVL    16(FRAME_PTR), PROGRAM_COUNTER          ; 0964
     E4          50       18  F3 000D4    AOBLEQ  #24, INDEX, 2$          ; 0951
   04 000D8 3$:   RET                   ; 0967

; Routine Size:  217 bytes,    Routine Base:  CODE + 056F
```

```
;   0968  1 END     ! End of CRDMMACT module
;   0969  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          463  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE         1608  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  WORK          500  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
Symbol                          Type  Defined  Referenced ...
---- ------------------------ -------- ----- ---------------
$ASCIC    Unbound           142     143
    Macro    Lib02   131      132      294      295      414      417      592      598      600
$BITPOSITION   Macro    Lib05   344
$CHECK_OA_STATUS  Macro      137   294      295      414      417      592      598      600
$CLOSE    KeyWMacr Lib05   597
$CONNECT   KeyWMacr Lib05   416
$CRD_LITERALS   Macro    Lib03   107
$CRD_PRINT   Unbound          141      145      146
    Macro    Lib03   194      253      294      295      414      417      592      598      600      632
$CRD_RETURN_LENGTH_STATUS Macro    Lib03   273      392      472      550      584
$DS_DSADEF   Macro    Lib02   106
$DS_TYPEDEF   Macro    Lib02   194      253      294      295      414      417      592      598      600      632
$EQULST    Macro    Lib05   107      108      194      253      294      295      414      417      592      598
            600      632
$ER    Comptime   128
$ERASE    KeyWMacr Lib05   599
$FAB    KeyWMacr Lib05   342
$FAB_DECL   Macro    Lib05   344
$FAO    Macro    Lib05   464
$GET    KeyWMacr Lib05   564
$LOAD_ASCID_A   KeyWMacr   161   460
$LOAD_ASCID_S   KeyWMacr   153   289      290      425      431      437      443      449
$MODULE    Bind        128
$OPEN    KeyWMacr Lib05   413
$RAB    KeyWMacr Lib05   345
$RAB_DECL   Macro    Lib05   345
$RAB_DECL_ASYN   Unbound          345
$RMS_BITFLD   Macro    Lib05   344      345
$RMS_BITS   Unbound          344      345
    Macro    Lib05   344
$RMS_CODFLD   Macro    Lib05   344      345
$RMS_OR    Macro    Lib05   344
$RMS_VALFLD   Unbound          344
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Macro    Lib05   344     345
$SCREEN_L TERALS  Macro    Lib03   108
ADDRESS    RoutForm   799   823.
    RoutForm   859   909.
ADR    KeyWForm   161   166
ARG    MacrForm   137   138     143     144
AUTO$K_EXIT_AUTOSIZER_ERROR Literal    107
AUTO$K_EXIT_CONTROL_C  Literal    107
AUTO$K_EXIT_DUMMY_2  Literal    107
AUTO$K_EXIT_DUMMY_3  Literal    107
AUTO$K_EXIT_ERRORS_COMPLETE Literal    107
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal    107
AUTO$K_EXIT_INTERNAL_ERROR Literal    107
AUTO$K_EXIT_INV_BASE_SYSTEM Literal    107
AUTO$K_EXIT_LAST_REASON  Literal    107   107
AUTO$K_EXIT_NOERRORS_COMPLETE Literal    107
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal    107
AUTO$K_EXIT_NOERRORS_PREP Literal    107
CLI_BUFFER_ADDRESS  RoutForm   200   270a=   272.
    RoutForm   302   389a=   391.    470a=   471.
    RoutForm   480   547a=   549.    574.    583.
CLI_BUFFER_LENGTH  RoutForm   200   270.    272.
    RoutForm   302   388.    391.    470.    471.
    RoutForm   480   547.    549.    574.    583.
CMD    Own        239   290=    293a
CNTRL    Own        524
CRD$AL_CRD_DISPATCH_VECTORS External Lib03   884m    897=    897.    926a
CRD$AUTOSIZER_ERROR  ExtRout  Lib03   853c
CRD$GA_DS_FLAGS   External Lib03   918a.
CRD$GA_FILE_FAB   Global     342   344=    406=    407=    407a.   408=    408.    410a    413a    514e
    Map        518   597a    599a
CRD$GA_FILE_RAB   Global     345   345=    410=    411=    416a    515e
    Map        519   564a
CRD$GA_HS_CURRENT_DDB_PTR External Lib03   920.
CRD$GA_HS_CURRENT_HSB_PTR External Lib03   921.
CRD$GA_REC_BUF   Global     341   411a    513e
    Map        517
CRD$GB_CURRENT_STATE_TABLE External Lib03   759=    911.
CRD$GL_CRD_DEBUG_VECTOR  External Lib03   903=    905=    907=    908=    909=    911=    912=    913=    914=    915=
            917=    918=    920=    921=    922=    924=    926=    945=    953=
CRD$GL_HS_CURRENT_HSB_NUMB External Lib03   922.
CRD$GL_HS_CURRENT_STATE  External Lib03   914.
CRD$GL_MM_CURRENT_STATE  External Lib03   578=    912.
CRD$GL_PREVIOUS_STATE  External Lib03   578.    915.
CRD$GL_TQ_CURRENT_STATE  External Lib03   913.
CRD$GL_TQ_CURRENT_TQ_ENTRY External Lib03   924.
CRD$GT_AUTOSIZER_ERROR  Unbound          146
    External Lib03   294a    295a    414a    417a    592a    598a    600a
CRD$GT_AUTOSIZER_NAME  External Lib03
    GlobBind   131   244m    268.
CRD$GT_AUTOSIZER_SET_FLAGS External Lib03
    GlobBind   132   363m    377.    385.    386.
CRD$GT_DS_LOAD_COM  External Lib03   245m    267.
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
CRD$GT_DS_SET_FLAGS_COM  External Lib03   364m    383.    384.
CRD$GT_DS_START_COM  External Lib03   531m    545.
CRD$K_ACTION_EQL_DO_NOTHING Literal    107
CRD$K_ACTION_RANGE_ERROR Literal    107
CRD$K_ADVANCE_DIAGNOSTIC Literal    107
CRD$K_ADVANCE_GENERAL_COM Literal    107
CRD$K_ADVANCE_STATE  Literal    107
CRD$K_ALLOCATION_ERROR  Literal    107
CRD$K_ASSIGN_PTABLE_PTRS Literal    107
CRD$K_AUTOSIZER_ERROR  Literal    107
CRD$K_AUTOSIZER_SET_FLAGS Literal    107
CRD$K_BAD_ACTION_NUMBER  Literal    107
CRD$K_BAD_TYPECODE_ERROR Literal    107
CRD$K_BASE_SYSTEM_IDC  Literal    107
CRD$K_BASE_SYSTEM_LESI  Literal    107
CRD$K_BASE_SYSTEM_NULL  Literal    107
CRD$K_BASE_SYSTEM_UDA_0  Literal    107
CRD$K_BASE_SYSTEM_UDA_1  Literal    107
CRD$K_BASE_SYSTEM_UDA_2  Literal    107
CRD$K_BASE_SYSTEM_UDA_3  Literal    107
CRD$K_CRD_INITIALIZATION Literal    107
CRD$K_CREATE_HST  Literal    107
CRD$K_DATA_LENGTH_ERROR  Literal    107
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal    107
CRD$K_DDB_BLINK  Literal    107
CRD$K_DDB_FLINK  Literal    107
CRD$K_DDB_LAST_ENTRY  Literal    107
CRD$K_DDB_MEMORY_SIZE  Literal    107
CRD$K_DDB_MENU_DEVICE_NUMB Literal    107
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal    107
CRD$K_DDB_MENU_SUPPORT_SIZE Literal    107
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal    107
CRD$K_DDB_PTABLE_ENTRY  Literal    107
CRD$K_DDB_STATUS  Literal    107
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal    107
CRD$K_DEVICE_NAME  Literal    107
CRD$K_DIAGNOSTIC_ERROR  Literal    107
CRD$K_DIAGNOSTIC_NAME  Literal    107
CRD$K_DONE_GENERAL_COMS  Literal    107
CRD$K_DO_NOTHING  Literal    107
CRD$K_DS_COMMAND_SIZE  Literal  Lib03   273     392     472     550     584
CRD$K_DUMMY_10  Literal    107
CRD$K_DUMMY_11  Literal    107
CRD$K_DUMMY_12  Literal    107
CRD$K_DUMMY_13  Literal    107
CRD$K_DUMMY_3  Literal    107
CRD$K_DUMMY_4  Literal    107
CRD$K_DUMMY_5  Literal    107
CRD$K_DUMMY_6  Literal    107
CRD$K_DUMMY_7  Literal    107
CRD$K_DUMMY_8  Literal    107
CRD$K_DUMMY_9  Literal    107
CRD$K_EXIT_CRD  Literal    107
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
CRD$K_FAILURE   Unbound          148
    Literal  Lib03   294     295     414     417     592     598     600     604
CRD$K_HOOK_POINT  Literal    107
CRD$K_HS_INTERNAL_ERROR  Literal    107
CRD$K_IDC_EVDLB   Literal    107
CRD$K_IDC_EVDLC   Literal    107
CRD$K_IDC_EVDLD   Literal    107
CRD$K_IDC_EVRAD_0  Literal    107
CRD$K_IDC_EVRAD_1  Literal    107
CRD$K_IDC_LAST_DIAGNOSTIC Literal    107
CRD$K_INTERNAL_ERROR  Literal    107
CRD$K_LAST_ACTION_ROUTINE Literal    107
CRD$K_LAST_ENTRY  Literal    107
CRD$K_LAST_FUNCTION  Literal    107
CRD$K_LAST_HOOK_POINT  Literal    107
CRD$K_LAST_INTERNAL_ERROR Literal    107
CRD$K_LAST_TYPE   Literal    107
CRD$K_LESI_ENWCA  Literal    107
CRD$K_LESI_EVRMB_0  Literal    107
CRD$K_LESI_EVRMB_1  Literal    107
CRD$K_LESI_LAST_DIAGNOSTIC Literal    107
CRD$K_LEVEL_4_PASSED  Literal    107
CRD$K_LOAD_AUTOSIZER  Literal    107
CRD$K_LOAD_ERROR  Literal    107
CRD$K_LOAD_GENERAL_COM  Literal    107
CRD$K_LOAD_LOAD_COM  Literal    107
CRD$K_LOAD_SELECT_COM  Literal    107
CRD$K_LOAD_SET_EVENT_COM Literal    107
CRD$K_LOAD_SET_FLAGS_COM Literal    107
CRD$K_LOAD_START_COM  Literal    107
CRD$K_LOAD_SUP_FILE  Literal    107
CRD$K_MM_INTERNAL_ERROR  Literal    107   823
CRD$K_NEW_LINE   Literal    107
CRD$K_NUMB_DISPATCH_VECTORS Literal  Lib03   891
CRD$K_NUMB_UNUSED_VECTORS Literal  Lib03   891
CRD$K_PREPARATION_ERROR  Literal    107
CRD$K_PREPARATION_ERROR_TEXT Literal    107
CRD$K_REPEAT_TURING  Literal  Lib03   196     379     643     673     701     731     761
CRD$K_RETURN_FAILURE  Literal  Lib03   795     855
CRD$K_RETURN_SUCCESS  Literal  Lib03   825
CRD$K_RUNTIME   Literal    107
CRD$K_SAME_LINE   Literal    107
CRD$K_SET_EVENT_ARGS  Literal    107
CRD$K_SET_FLAGS_ARGS  Literal    107
CRD$K_SET_UP_DIAGNOSTIC  Literal    107
CRD$K_START   Literal    107
CRD$K_START_AUTOSIZER  Literal    107
CRD$K_START_ERROR  Literal    107
CRD$K_START_QUALIFIERS  Literal    107
CRD$K_STAR_LINE   Literal    107
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal    107
CRD$K_STATE_ADVANCE_GENERAL_COM Literal    107
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal    107
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
CRD$K_STATE_AUTOSIZER_ERROR Literal    107
CRD$K_STATE_AUTOSIZER_RUNNING Literal    107
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal    107
CRD$K_STATE_DIAGNOSTIC_ERROR Literal    107
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal    107
CRD$K_STATE_DONE_DIAGNOSTICS Literal    107
CRD$K_STATE_DONE_GENERAL_COMS Literal    107
CRD$K_STATE_DUMMY_1  Literal    107
CRD$K_STATE_DUMMY_10  Literal    107
CRD$K_STATE_DUMMY_12  Literal    107
CRD$K_STATE_DUMMY_13  Literal    107
CRD$K_STATE_DUMMY_2  Literal    107
CRD$K_STATE_DUMMY_3  Literal    107
CRD$K_STATE_DUMMY_4  Literal    107
CRD$K_STATE_DUMMY_6  Literal    107
CRD$K_STATE_DUMMY_7  Literal    107
CRD$K_STATE_DUMMY_8  Literal    107
CRD$K_STATE_EXIT_CRD  Literal    107
CRD$K_STATE_INTERNAL_ERROR Literal    107
CRD$K_STATE_LAST_STATE  Literal    107
CRD$K_STATE_LEVEL_4_PASSED Literal    107
CRD$K_STATE_LOAD_AUTOSIZER Literal    107
CRD$K_STATE_LOAD_ERROR  Literal    107
CRD$K_STATE_LOAD_GENERAL_COM Literal    107
CRD$K_STATE_LOAD_LOAD_COM Literal    107
CRD$K_STATE_LOAD_SELECT_COM Literal    107
CRD$K_STATE_LOAD_SET_EVENT_COM Literal    107
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal    107
CRD$K_STATE_LOAD_START_COM Literal    107
CRD$K_STATE_PREPARATION_ERROR Literal    107
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal    107
CRD$K_STATE_START  Literal    107
CRD$K_STATE_START_AUTOSIZER Literal    107
CRD$K_STATE_START_ERROR  Literal    107
CRD$K_SUCCESS   Unbound          273     392     472     550     584
    Literal  Lib03   273     298     392     455     465     472     474     550     584     587
             601
CRD$K_TEST_START_TEXT  Literal    107
CRD$K_TOTAL_NUMB_VECTORS Literal  Lib03   891     903     944     951
CRD$K_TQ_INTERNAL_ERROR  Literal    107
CRD$K_TYPECODE   Literal    107
CRD$K_UDA_0_EVDLB  Literal    107
CRD$K_UDA_0_EVDLC  Literal    107
CRD$K_UDA_0_EVDLD  Literal    107
CRD$K_UDA_0_EVRLA_0  Literal    107
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal    107
CRD$K_UDA_1_EVDLB  Literal    107
CRD$K_UDA_1_EVDLC  Literal    107
CRD$K_UDA_1_EVDLD  Literal    107
CRD$K_UDA_1_EVRLA_0  Literal    107
CRD$K_UDA_1_EVRLA_1  Literal    107
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal    107
CRD$K_UDA_2_EVDLB  Literal    107
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_UDA_2_EVDLC  Literal    107
CRD$K_UDA_2_EVDLD  Literal    107
CRD$K_UDA_2_EVRLA_0  Literal    107
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal    107
CRD$K_UDA_3_EVDLB  Literal    107
CRD$K_UDA_3_EVDLC  Literal    107
CRD$K_UDA_3_EVDLD  Literal    107
CRD$K_UDA_3_EVRLA_0  Literal    107
CRD$K_UDA_3_EVRLA_1  Literal    107
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal    107
CRD$PRINT   External Lib03   194ac   253ac   294ac   295ac   414ac   417ac   592ac   598ac   600ac   632ac
CRD$PRINT_DS_COMMAND  ExtRout  Lib03   272c    391c    471c    549c    583c
CRD$SCREEN_PAUSE  ExtRout  Lib03   635c
CRD$STOP   Unbound          147
    ExtRout  Lib03   294c    295c    414c    417c    592c    598c    600c    824c    854c
CRD_ASSERT   Macro    Lib03   890
DS$K_PRINTB   Literal    194
    Literal    253
    Literal    294
    Literal    294
    Literal    294
    Literal    295
    Literal    295
    Literal    295
    Literal    414
    Literal    414
    Literal    414
    Literal    414
    Literal    417
    Literal    417
    Literal    417
    Literal    592
    Literal    592
    Literal    592
    Literal    598
    Literal    598
    Literal    598
    Literal    600
    Literal    600
    Literal    600
    Literal    632
DS$K_PRINTF   Literal    194   194
    Literal    253   253
    Literal    294   294
    Literal    294   294
    Literal    294   294
    Literal    295   295
    Literal    295   295
    Literal    295   295
    Literal    414   414
    Literal    414   414
    Literal    414   414
    Literal    417   417
    Literal    417   417
```

Symbol	Type	Defined	Referenced ...
	Literal	417	417
	Literal	592	592
	Literal	592	592
	Literal	592	592
	Literal	598	598
	Literal	598	598
	Literal	598	598
	Literal	600	600
	Literal	600	600
	Literal	600	600
	Literal	632	632
DS$K_PRINTI	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_PRINTX	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	

Symbol	Type	Defined	Referenced ...
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_ABORT_PROGRAM	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_ABORT_TEST	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    632
DS$K_TYPE_COMMAND_ERR  Literal    194
    Literal    253
    Literal    294
    Literal    294
    Literal    294
    Literal    295
    Literal    295
    Literal    295
    Literal    414
    Literal    414
    Literal    414
    Literal    417
    Literal    417
    Literal    417
    Literal    592
    Literal    592
    Literal    592
    Literal    598
    Literal    598
    Literal    598
    Literal    600
    Literal    600
    Literal    600
    Literal    632
DS$K_TYPE_COMMAND_OUT  Literal    194
    Literal    253
    Literal    294
    Literal    294
    Literal    294
    Literal    295
    Literal    295
    Literal    295
    Literal    414
    Literal    414
    Literal    414
    Literal    417
    Literal    417
    Literal    417
    Literal    592
    Literal    592
    Literal    592
    Literal    598
    Literal    598
    Literal    598
    Literal    600
    Literal    600
    Literal    600
    Literal    632
DS$K_TYPE_CRD_AUTOTEST  Literal    194   194
    Literal    253   253
    Literal    294   294
    Literal    294   294
```

Symbol	Type	Defined	Referenced ...
	Literal	294	294
	Literal	295	295
	Literal	295	295
	Literal	295	295
	Literal	414	414
	Literal	414	414
	Literal	414	414
	Literal	417	417
	Literal	417	417
	Literal	417	417
	Literal	592	592
	Literal	592	592
	Literal	592	592
	Literal	598	598
	Literal	598	598
	Literal	598	598
	Literal	600	600
	Literal	600	600
	Literal	600	600
	Literal	632	632
DS$K_TYPE_DS_PROMPT	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_DS_START	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	

Symbol	Type	Defined	Referenced ...
	Literal		414
	Literal		414
	Literal		417
	Literal		417
	Literal		417
	Literal		592
	Literal		592
	Literal		592
	Literal		598
	Literal		598
	Literal		598
	Literal		600
	Literal		600
	Literal		600
	Literal		632
DS$K_TYPE_ERRDEV	Literal		194
	Literal		253
	Literal		294
	Literal		294
	Literal		294
	Literal		295
	Literal		295
	Literal		295
	Literal		414
	Literal		414
	Literal		414
	Literal		417
	Literal		417
	Literal		417
	Literal		592
	Literal		592
	Literal		592
	Literal		598
	Literal		598
	Literal		598
	Literal		600
	Literal		600
	Literal		600
	Literal		632
DS$K_TYPE_ERRHARD	Literal		194
	Literal		253
	Literal		294
	Literal		294
	Literal		294
	Literal		295
	Literal		295
	Literal		295
	Literal		414
	Literal		414
	Literal		414
	Literal		417
	Literal		417
	Literal		417

Symbol	Type	Defined	Referenced ...
	Literal 592		
	Literal 592		
	Literal 592		
	Literal 598		
	Literal 598		
	Literal 598		
	Literal 600		
	Literal 600		
	Literal 600		
	Literal 632		
DS$K_TYPE_ERROR_BODY	Literal 194		
	Literal 253		
	Literal 294		
	Literal 294		
	Literal 294		
	Literal 295		
	Literal 295		
	Literal 295		
	Literal 414		
	Literal 414		
	Literal 414		
	Literal 417		
	Literal 417		
	Literal 417		
	Literal 592		
	Literal 592		
	Literal 592		
	Literal 592		
	Literal 598		
	Literal 598		
	Literal 598		
	Literal 600		
	Literal 600		
	Literal 600		
	Literal 632		
DS$K_TYPE_ERROR_END	Literal 194		
	Literal 253		
	Literal 294		
	Literal 294		
	Literal 294		
	Literal 295		
	Literal 295		
	Literal 295		
	Literal 414		
	Literal 414		
	Literal 414		
	Literal 417		
	Literal 417		
	Literal 417		
	Literal 592		
	Literal 592		
	Literal 592		
	Literal 598		
	Literal 598		

Symbol	Type	Defined	Referenced ...
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_ERRPREP	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_ERRSOFT	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DS$K_TYPE_ERRSUP  Literal    194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_ERRSYS  Literal    194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_ERR_HALT  Literal    194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
```

Symbol   Type Defined Referenced ...
------------------------------ -------- ----- ---------------

    Literal   295
    Literal   295
    Literal   295
    Literal   414
    Literal   414
    Literal   414
    Literal   417
    Literal   417
    Literal   417
    Literal   592
    Literal   592
    Literal   592
    Literal   598
    Literal   598
    Literal   598
    Literal   600
    Literal   600
    Literal   600
    Literal   632
DS$K_TYPE_EXCEPTION Literal   194
    Literal   253
    Literal   294
    Literal   294
    Literal   294
    Literal   295
    Literal   295
    Literal   295
    Literal   295
    Literal   414
    Literal   414
    Literal   414
    Literal   417
    Literal   417
    Literal   417
    Literal   592
    Literal   592
    Literal   592
    Literal   598
    Literal   598
    Literal   598
    Literal   600
    Literal   600
    Literal   600
    Literal   632
DS$K_TYPE_EXCEPTION_HEAD Literal   194
    Literal   253
    Literal   294
    Literal   294
    Literal   294
    Literal   295
    Literal   295
    Literal   295
    Literal   414
    Literal   414

Symbol	Type	Defined	Referenced ...
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_FIRST_PASS	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_GENERAL	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	

Symbol	Type	Defined	Referenced ...
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_GENERAL_ERROR	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_NO_TESTS	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	

Symbol	Type	Defined	Referenced ...
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_PARAM_ERROR	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_PROGRAM_END	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_PROGRAM_INFO	Literal	194	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_PROGRAM_START  Literal    194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_QIO_INVADP  Literal    194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
```

Symbol	Type	Defined	Referenced ...
Literal	295		
Literal	295		
Literal	414		
Literal	414		
Literal	414		
Literal	417		
Literal	417		
Literal	417		
Literal	592		
Literal	592		
Literal	592		
Literal	598		
Literal	598		
Literal	598		
Literal	600		
Literal	600		
Literal	600		
Literal	632		
DS$K_TYPE_QIO_NODRIVER	Literal	194	
Literal	253		
Literal	294		
Literal	294		
Literal	294		
Literal	295		
Literal	295		
Literal	295		
Literal	295		
Literal	414		
Literal	414		
Literal	414		
Literal	417		
Literal	417		
Literal	417		
Literal	592		
Literal	592		
Literal	592		
Literal	598		
Literal	598		
Literal	598		
Literal	600		
Literal	600		
Literal	600		
Literal	632		
DS$K_TYPE_QIO_WRONGVER	Literal	194	
Literal	253		
Literal	294		
Literal	294		
Literal	294		
Literal	295		
Literal	295		
Literal	295		
Literal	414		
Literal	414		
Literal	414		

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_SCRIPT_ECHO  Literal     194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_SCRIPT_PNF  Literal     194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
```

Symbol	Type	Defined	Referenced ...
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_SCRIPT_PROMPT	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	
	Literal	600	
	Literal	600	
	Literal	632	
DS$K_TYPE_SCRIPT_SKIP	Literal	194	
	Literal	253	
	Literal	294	
	Literal	294	
	Literal	294	
	Literal	295	
	Literal	295	
	Literal	295	
	Literal	414	
	Literal	414	
	Literal	414	
	Literal	417	
	Literal	417	
	Literal	417	
	Literal	592	
	Literal	592	
	Literal	592	
	Literal	598	
	Literal	598	
	Literal	598	
	Literal	600	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_SEQUENCE_ERROR Literal     194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_START_ERR  Literal     194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_START_LIST  Literal     194
    Literal     253
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_SUMMARY  Literal     194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DS$K_TYPE_USER_PROMPT  Literal     194
    Literal     253
    Literal     294
    Literal     294
    Literal     294
    Literal     295
    Literal     295
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     295
    Literal     414
    Literal     414
    Literal     414
    Literal     417
    Literal     417
    Literal     417
    Literal     592
    Literal     592
    Literal     592
    Literal     598
    Literal     598
    Literal     598
    Literal     600
    Literal     600
    Literal     600
    Literal     632
DSA$AL_APTMAIL   Bind        106   106
DSA$GL_APTCOM   Bind        106
DSA$GL_DEVLEN   Bind        106
DSA$GL_DEVNAM   Bind        106
DSA$GL_ERRNO   Bind        106
DSA$GL_EVENT   Bind        106
DSA$GL_FLAGS   Bind        106   263     284     294     295     374     403     414     417     541     560
             592     598     600     729     917
DSA$GL_MSGPTR   Bind        106
DSA$GL_MSGTYP   Bind        106
DSA$GL_PASSES   Bind        106
DSA$GL_PASSNO   Bind        106
DSA$GL_SECTNO   Bind        106
DSA$GL_SID   Bind        106   422
DSA$GL_SUBTNO   Bind        106
DSA$GL_TESTNO   Bind        106
DSA$GL_UNITS   Bind        106
DSA$V_CRD_MENUTEST_OFF  Macro    Lib02   263     374     541
DSA$V_CRD_MENUTEST_ON  Macro    Lib02   284     403     560
DSA$V_CRD_TRACE   Unbound          140
    Macro    Lib02   294     295     414     417     592     598     600
DSA$V_OPER   Macro    Lib02   729
DSC$K_CLASS_S   Unbound          157     165
    Literal  Lib05   289     290     429     435     441     447     453     460
DSC$K_DTYPE_T   Unbound          156     164
    Literal  Lib05   289     290     429     435     441     447     453     460
ERROR_TYPE   RoutForm   859   905.
EVSBB$S_EVSBB$REC  Literal  Lib04   341     345
FAB$B_FNS   Macro    Lib05   407
FAB$C_BID   Literal  Lib05   344
FAB$C_BLN   Literal  Lib05   344
FAB$C_SEQ   Literal  Lib05   344
FAB$C_VAR   Literal  Lib05   344
FAB$L_FNA   Macro    Lib05   406     407     408
FAB$M_CR   Literal  Lib05   344
FAB$M_GET   Literal  Lib05   344
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
FAB$V_CHAN_MODE    Macro    Lib05   344
FAB$V_FILE_MODE    Macro    Lib05   344
FAB$V_LNM_MODE     Macro    Lib05   344
FAO_BUFF    Own          357    460a    469.
FAO_BUFF_SIZE    Literal    354   357     460
FAO_INPUT_DSC    Own          359    429=    435=    441=    447=    453=    464a
FAO_LENGTH    Own          358    464a    469.
FAO_OUTPUT_DSC    Own          360    460=    464a
FP    Builtin    938    947.
FRAME_PTR    Local      942    947=    956=    956.    958.    961.    964.
GET1ST_     Macro    Lib05   107     108     194     253     294     295     414     417     592     598
              600     632
GET2ND_     Unbound          107     108     194     253     294     295     414     417     592     598
              600     632
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal    107
HS$K_ACTION_ADVANCE_GENERAL_COM Literal    107
HS$K_ACTION_ADVANCE_STATE Literal    107
HS$K_ACTION_CHANGE_TABLE Literal    107
HS$K_ACTION_DIAGNOSTIC_ERROR Literal    107
HS$K_ACTION_DONE_GENERAL_COMS Literal    107
HS$K_ACTION_DO_NOTHING  Literal    107
HS$K_ACTION_DUMMY_3  Literal    107
HS$K_ACTION_DUMMY_4  Literal    107
HS$K_ACTION_DUMMY_5  Literal    107
HS$K_ACTION_DUMMY_6  Literal    107
HS$K_ACTION_INIT_HS  Literal    107
HS$K_ACTION_INTERNAL_ERROR Literal    107
HS$K_ACTION_LAST_ACTION  Literal    107
HS$K_ACTION_LOAD_ERROR  Literal    107
HS$K_ACTION_LOAD_GENERAL_COM Literal    107
HS$K_ACTION_LOAD_LOAD_COM Literal    107
HS$K_ACTION_LOAD_SELECT_COM Literal    107
HS$K_ACTION_LOAD_SET_EVENT_COM Literal    107
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal    107
HS$K_ACTION_LOAD_START_COM Literal    107
HS$K_ACTION_PREPARATION_ERROR Literal    107
HS$K_ACTION_START_ERROR  Literal    107
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal    107
HS$K_STATE_ADVANCE_GENERAL_COM Literal    107
HS$K_STATE_CHANGE_TABLE  Literal    107
HS$K_STATE_DIAGNOSTIC_ERROR Literal    107
HS$K_STATE_DIAGNOSTIC_RUNNING Literal    107
HS$K_STATE_DONE_GENERAL_COMS Literal    107
HS$K_STATE_DUMMY_1  Literal    107
HS$K_STATE_DUMMY_2  Literal    107
HS$K_STATE_DUMMY_3  Literal    107
HS$K_STATE_DUMMY_4  Literal    107
HS$K_STATE_DUMMY_6  Literal    107
HS$K_STATE_INIT_HS  Literal    107
HS$K_STATE_INTERNAL_ERROR Literal    107
HS$K_STATE_LAST_STATE  Literal    107
HS$K_STATE_LOAD_ERROR  Literal    107
HS$K_STATE_LOAD_GENERAL_COM Literal    107
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
HS$K_STATE_LOAD_LOAD_COM Literal    107
HS$K_STATE_LOAD_SELECT_COM Literal    107
HS$K_STATE_LOAD_SET_EVENT_COM Literal    107
HS$K_STATE_LOAD_SET_FLAGS_COM Literal    107
HS$K_STATE_LOAD_START_COM Literal    107
HS$K_STATE_PREPARATION_ERROR Literal    107
HS$K_STATE_START_ERROR  Literal    107
INDEX    Local      944   945.
    Local      951   953.
IOSTATUS   Own        241   293=    294.
LENGTH    RoutForm   799   823.
    RoutForm   859   908.
LIB$SPAWN   ExtRout    236   293c
MEN$BUILD_DDB   ExtRout  Lib03   640c
MEN$DETERMINE_SUPPORT  ExtRout  Lib03   671c
MEN$FUNCTEST_MENU  ExtRout  Lib03   730c
MEN$GENERATE_ONLINE_ATTACH ExtRout     80   573c
MEN$GT_AUTOSIZER_BEGIN  External Lib03   253a
MEN$GT_AUTOSIZER_END  External Lib03   632a
MEN$GT_FAILED_ONL_AUTOSIZER Unbound          145
    External Lib03   294a    295a    414a    417a    592a    598a    600a
MEN$GT_MENUTEST_BEGIN  External Lib03   194a
MEN$K_HS_STATE_TABLE  Literal    107
MEN$K_MM_STATE_TABLE  Literal    107
MEN$K_TQ_STATE_TABLE  Literal    107   759
MEN$MENU_TEST_INTERNAL_ERROR GlobRout   859 Lib03e    100f    823c    948a
MEN$SPECIFY_SELECTNO  ExtRout  Lib03   672c
MENU$K_EXIT_AUTOSIZER_ERROR Unbound          147
    Literal    107   294     295     414     417     592     598     600     854
MENU$K_EXIT_INTERNAL_ERROR Literal    107   824
MENU$K_EXIT_LAST_REASON  Literal    107
MENU$K_EXIT_OPERATOR_ABORT Literal    107
MENU$K_EXIT_OPERATOR_STOP Literal    107
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal    107
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal    107
MM$AUTOSIZER_ERROR  GlobRout   829 Lib03e     98f
MM$BUILD_DDBS   GlobRout   608 Lib03e     88f
MM$BUILD_HSTS   GlobRout   647 Lib03e     89f
MM$CHANGE_TABLE   GlobRout   735 Lib03e     94f
MM$DONE_INITS   GlobRout   677 Lib03e     91f
MM$INTERNAL_ERROR  GlobRout   799 Lib03e     97f
MM$K_ACTION_ADVANCE_STATE Literal    107
MM$K_ACTION_AUTOSIZER_ERROR Literal    107
MM$K_ACTION_BUILD_DDBS  Literal    107
MM$K_ACTION_BUILD_HSTS  Literal    107
MM$K_ACTION_CHANGE_TABLE Literal    107
MM$K_ACTION_DONE_INITS  Literal    107
MM$K_ACTION_DO_NOTHING  Literal    107
MM$K_ACTION_DUMMY_3  Literal    107
MM$K_ACTION_DUMMY_4  Literal    107
MM$K_ACTION_DUMMY_5  Literal    107
MM$K_ACTION_DUMMY_6  Literal    107
MM$K_ACTION_DUMMY_7  Literal    107
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_ACTION_DUMMY_8  Literal    107
MM$K_ACTION_INTERNAL_ERROR Literal    107
MM$K_ACTION_LAST_ACTION  Literal    107
MM$K_ACTION_LOAD_AUTOSIZER Literal    107
MM$K_ACTION_MENU_PROMPT  Literal    107
MM$K_ACTION_PRINT_MENU  Literal    107
MM$K_ACTION_PRINT_MENU_HEADING Literal    107
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal    107
MM$K_ACTION_START_AUTOSIZER Literal    107
MM$K_STATE_AUTOSIZER_ERROR Literal    107
MM$K_STATE_AUTOSIZER_RUNNING Literal    107
MM$K_STATE_BUILD_DDBS  Literal    107
MM$K_STATE_BUILD_HSTS  Literal    107
MM$K_STATE_CHANGE_TABLE  Literal    107
MM$K_STATE_DONE_INITS  Literal    107
MM$K_STATE_DUMMY_1  Literal    107
MM$K_STATE_DUMMY_2  Literal    107
MM$K_STATE_DUMMY_3  Literal    107
MM$K_STATE_DUMMY_5  Literal    107
MM$K_STATE_DUMMY_7  Literal    107
MM$K_STATE_DUMMY_8  Literal    107
MM$K_STATE_INTERNAL_ERROR Literal    107
MM$K_STATE_LAST_STATE  Literal    107
MM$K_STATE_LOAD_AUTOSIZER Literal    107
MM$K_STATE_MENU_PROMPT  Literal    107
MM$K_STATE_PRINT_MENU  Literal    107
MM$K_STATE_PRINT_MENU_HEADING Literal    107
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal    107
MM$K_STATE_START  Literal    107
MM$K_STATE_START_AUTOSIZER Literal    107
MM$LOAD_AUTOSIZER  GlobRout   200 Lib03e     84f
MM$MENU_PROMPT   GlobRout   765 Lib03e     95f
MM$PRINT_MENU   GlobRout   705 Lib03e     93f
MM$PRINT_MENU_HEADING  GlobRout   170 Lib03e     83f
MM$SET_FLAGS_AUTOSIZER  GlobRout   302 Lib03e     85f
MM$START_AUTOSIZER  GlobRout   480 Lib03e     86f
MODNAM    Macro    Lib01   128
NAM    KeyWForm   153   155     156     157     158
    KeyWForm   161   163     164     165     166
NUL2ND_    Macro    Lib05   107     108     194     253     294     295     414     417     592     598
           600     632
OUT    Own        240   289=    293a
OUTPUT_BUFF    Own        528
OUTPUT_DESC    Own        525   525=
OUTPUT_LEN   Own        526
OUTPUT_SIZE   Literal    522   525     528
PC    Builtin    937
PR$_SID_TYP730   Literal  Lib05   448
PR$_SID_TYP750   Literal  Lib05   442
PR$_SID_TYP780   Literal  Lib05   436
PR$_SID_TYP790   Literal  Lib05   430
PR$_SID_TYP8SS   Literal  Lib05   424
PROBEW    Builtin    936   961
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
PROGRAM_COUNTER   Local       941   948=    953.    964=
RAB$C_BID   Literal  Lib05   345
RAB$C_BLN   Literal  Lib05   345
RAB$C_SEQ   Literal  Lib05   345
RAB$L_FAB   Macro    Lib05   410
RAB$L_UBF   Macro    Lib05   411
RMS$_EOF   Literal  Lib05   567
RMS$_NORMAL   Literal  Lib05   570
SCREEN$K_CLEAR_SCREEN  Literal    108
SCREEN$K_COUNT_LINE  Literal    108
SCREEN$K_PRESS_RETURN  Literal    108   635
SCREEN$K_PRESS_RETURN_MM Literal    108
SCREEN$K_PRESS_RETURN_TM Literal    108
SCREEN$K_TM_MSG   Literal    108
SDL$STARLET_CONCAT  Macro    Lib05   413     416     564     597     599
SDL$STARLET_OPT   Macro    Lib05   413     416     564     597     599
SDL$STARLET_REQ   Macro    Lib05   413     416     564     597     599
SIZ    KeyWForm   161   163
STATUS    Own         241   293a    295.
    Own         356   413=    414.    416=    417.    464=    465.
    Own         527   564=    565.    573=    581.    592.    597=    598.    600.
STR    KeyWForm   153   155     158
SYS$CLOSE   ExtRout    597   597c
SYS$CONNECT   ExtRout    416   416c
SYS$ERASE   ExtRout    599   599c
SYS$FAO    ExtRout    464   464c
SYS$GET    ExtRout    564   564c
SYS$OPEN   ExtRout    413   413c
TQ$K_ACTION_ADVANCE_STATE Literal    107
TQ$K_ACTION_CHANGE_TABLE Literal    107
TQ$K_ACTION_DONE_TEST_QUEUE Literal    107
TQ$K_ACTION_DO_NOTHING  Literal    107
TQ$K_ACTION_DUMMY_3  Literal    107
TQ$K_ACTION_DUMMY_4  Literal    107
TQ$K_ACTION_DUMMY_5  Literal    107
TQ$K_ACTION_GET_DDB  Literal    107
TQ$K_ACTION_INIT_TQ  Literal    107
TQ$K_ACTION_INTERNAL_ERROR Literal    107
TQ$K_ACTION_LAST_ACTION  Literal    107
TQ$K_STATE_CHANGE_TABLE  Literal    107
TQ$K_STATE_DONE_TEST_QUEUE Literal    107
TQ$K_STATE_DUMMY_1  Literal    107
TQ$K_STATE_DUMMY_2  Literal    107
TQ$K_STATE_DUMMY_3  Literal    107
TQ$K_STATE_DUMMY_4  Literal    107
TQ$K_STATE_DUMMY_5  Literal    107
TQ$K_STATE_GET_DDB  Literal    107
TQ$K_STATE_INIT_TQ  Literal    107
TQ$K_STATE_INTERNAL_ERROR Literal    107
TQ$K_STATE_LAST_STATE  Literal    107
TRUE    Literal  Lib03   729
TYPE    Own         527
TYPECODE   RoutForm   799   823.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    RoutForm   859   907.



  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDMMACT.B32;1
    2  Module CRDMMACT
   61  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   64  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   67  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   70  Library #4 DISK$DIAGPACK03:[DS.EVSBB.DEF]EVSBB_DEF.L32;1
   73  Library #5 SYS$COMMON:[SYSLIB]LIB.L32;2
  969  Eludom CRDMMACT


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     a Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;           --- -  - Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671         1          0          46          00:00.2
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923         7          0          94          00:00.2
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486        59         12          62          00:00.1
;  DISK$DIAGPACK03:[DS.EVSBB.DEF]EVSBB_DEF.L32;1
;              44         1          2           7          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619         47           0       1000          00:04.6
```

```
;     COMMAND QUALIFIERS

; BLISS CRDMMACT.B32/LIST=CRDMMACT.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDMMACT.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  1608 code + 963 data bytes
; Run Time:    01:15.8
; Elapsed Time:    02:55.1
; Lines/CPU Min:     767
; Lexemes/CPU-Min: 68099
; Memory Used: 385 pages
; Compilation Complete
```

```
;    0001   0 %TITLE '*** CRDPTABLE Module'
;    0002   0 MODULE CRDPTABLE (    ! Acces the Ptables
;    0003   0   IDENT = '01-01'
;    0004   0   ) =
;    0005   0 !++
;    0006   0 ! COPYRIGHT (c) 1980, 1981
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the CRD-Loadable image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! Access the Resident-DS's PTables.
;    0032  0 !
;    0033  0 ! Author:
;    0034  0 !
;    0035  0 ! Jack Stansbury
;    0036  0 !
;    0037  0 ! Creation Date:
;    0038  0 !
;    0039  0 ! May 4, 1982
;    0040  0 !
;    0041  0 ! Version:
;    0042  0 !
;    0043  0 ! 6.7
;    0044  0 !
;    0045  0 ! Modified By:
;    0046  0 !
;    0047  0 ! 01 Jack Stansbury, Version 6.9, August 18, 1982
;    0048  0 !  Added the CRD$Get_Device_Name routine that will return a pointer to
;    0049  0 !  an ASCIC device name string, given a pointer to a PTable.
;    0050  0 !--
```

```
;    0051  0 %SBTTL 'Libraries'
;    0052  1 BEGIN     ! Begin the CRDPTABLE module
;    0053  1
;    0054  1 LIBRARY
;    0055  1  '$CRD';     ! Use CRD.L32 first
;    0056  1
;    0057  1 LIBRARY
;    0058  1  '$DS';     ! Use Ds.L32 second
;    0059  1
;    0060  1 LIBRARY
;    0061  1  '$Diag';    ! Use Diag.L32 third
;    0062  1
;    0063  1 LIBRARY
;    0064  1  'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0065  1 %SBTTL 'Table of Contents'
;    0066  1
;    0067  1 FORWARD ROUTINE
;    0068  1  CRD$Find_PTable   : ADDRESSING_MODE (LONG_RELATIVE),
;    0069  1       ! Find a PTable containing a certain device name
;    0070  1
;    0071  1  CRD$Get_Device_Name  : ADDRESSING_MODE (LONG_RELATIVE),     ![01]
;    0072  1       ! Return a pointer to an ASCIC device name string   ![01]
;    0073  1
;    0074  1  CRD$Print_PTables  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
;    0075  1       ! Print information in the PTables
```

```
;    0076  1 %SBTTL 'Included Macros and Literals'
;    0077  1
;    0078  1  $CRD_Literals;    ! Define the CRD literals
```

```
;    0079  1 %SBTTL 'Psect Definitions'
;    0080  1
;    0081  1 PSECT
;    0082  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG RELATIVE));
;    0083  1
;    0084  1 PSECT
;    0085  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0086  1
;    0087  1 PSECT
;    0088  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0089  1
;    0090  1 PSECT
;    0091  1     GLOBAL - Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0092  1 %SBTTL 'Module-Global Static Storage'
;    0093  1
;    0094  1  ModNam ('CRDPTABLE');  ! Module name for ErrSup macro
```

```
;    0095  1 %SBTTL 'CRD$Find_PTable Routine'
;    0096  1
;    0097  1 GLOBAL ROUTINE CRD$Find_PTable (Device_Name) =
;    0098  1
;    0099  1 !++
;    0100  1 ! Functional Description:
;    0101  1 !
;    0102  1 ! Find a PTable containg a device called Device_Name.
;    0103  1 !
;    0104  1 ! Formal Input Parameters:
;    0105  1 !
;    0106  1 ! Device_Name: A pointer to an ASCIC string.
;    0107  1 !
;    0108  1 ! Formal Output Parameters:
;    0109  1 !
;    0110  1 ! None
;    0111  1 !
;    0112  1 ! Side Effects:
;    0113  1 !
;    0114  1 ! None
;    0115  1 !
;    0116  1 ! Completion Codes:
;    0117  1 !
;    0118  1 ! 0    No PTable was found containg this Device_Name.
;    0119  1 ! Address - Address contains the address of the PTable.
;    0120  1 !--
```

```
;    0121  2 BEGIN      ! Routine CRD$Find_PTable
;    0122  2  BIND
;    0123  2   Numb_PTable_Entries = (.CRD$GA_PTable),
;    0124  2   PTable_Table      = (.CRD$GA_PTable + 4) : VECTOR [, LONG];
;    0125  2
;    0126  2  LOCAL
;    0127  2   PTable_Table_Index, ! Index into the PTable_Table (table)
;    0128  2   PTable_Ptr,  ! A pointer to a ptable
;    0129  2   PTable_Device_Name, ! A pointer to an ASCII string
;    0130  2   PTable_Device_Name_Len, ! The length of the above ASCII string
;    0131  2   PTable_Device_Name_Desc;! A descriptor of a device name - in the PTable.
;    0132  2
;    0133  2  MAP
;    0134  2   Device_Name : REF VECTOR [, BYTE];
;    0135  2      ! Reference Device_Name as a vector of bytes
;    0136  2
;    0137  2  IF (.Device_Name [0] EQL 0) THEN
;    0138  2   !+
;    0139  2   ! The device name is null - i.e., no count byte. Return the 0 address.
;    0140  2   !-
;    0141  2   RETURN (0);
;    0142  2
;    0143  2  PTable_Table_Index = .Numb_PTable_Entries - 1;
;    0144  2      ! Start at the last PTable address
;    0145  2
;    0146  2  !+
;    0147  2  ! Search through the list of PTable addresses to find one that contains a device name
;    0148  2  ! equal to Device_Name. Keep looking until a zero PTable pointer is found. This signifies
;    0149  2  ! the end of the PTable entries. If a match is found, return a pointer to this PTable.
;    0150  2  !-
;    0151  2
;    0152  2  WHILE ((Ptable_Ptr = .PTable_Table [.PTable_Table_Index]) NEQ 0) DO
;    0153  3   BEGIN
;    0154  3   BIND
;    0155  3    PTable = (.PTable_Ptr) : $DS_HPO_DECL ();
;    0156  3
;    0157  3   !+
;    0158  3   ! Get the device name from the table and get rid of any leading underscores.
;    0159  3   !-
;    0160  3
;    0161  3   PTable_Device_Name_Desc = PTable [HP$Q_Device];
;    0162  4    BEGIN
;    0163  4    MAP
;    0164  4     PTable_Device_Name_Desc : REF VECTOR [2, LONG];
;    0165  4
;    0166  4    PTable_Device_Name_Len = .PTable_Device_Name_Desc [0];
;    0167  4    PTable_Device_Name     = .PTable_Device Name Desc [1];
;    0168  4
;    0169  5     BEGIN
;    0170  5     MAP
;    0171  5      PTable_Device_Name : REF VECTOR [, BYTE];
;    0172  5
;    0173  5     WHILE (.PTable_Device_Name [0] EQL %C'_') DO
;    0174  6      BEGIN
;    0175  6      PTable Device_Name     - PTable_Device_Name [1];
```

```
;    0176  6      PTable_Device_Name_Len = .PTable_Device_Name_Len - 1;
;    0177  5      END;
;    0178  4     END;
;    0179  3    END;
;    0180  3
;    0181  3    !+
;    0182  3    ! Now, Ptable_Device_Name     = address of the device name string in the PTable
;    0183  3    !      PTable_Device_Name_Len = length of the device name string in the PTable
;    0184  3    ! Compare the two strings. If equal, return address. If not, keep looking.
;    0185  3    !-
;    0186  3
;    0187  4    IF (CH$EQL (.PTable_Device_Name_Len, CH$PTR (.PTable_Device_Name),
;    0188  4        .Device_Name [0],      CH$PTR (Device_Name [1])))
;    0189  3    THEN
;    0190  4     RETURN (.PTable_Ptr)
;    0191  3    ELSE
;    0192  3     PTable_Table_Index = .PTable_Table_Index - 1;
;    0193  3       ! Check the next ptable pointer
;    0194  2    END;
;    0195  2
;    0196  2  RETURN (0);    ! Return a 0 address
;    0197  1 END;      ! Routine CRD$Find_PTable


            .TITLE  CRDPTABLE *** CRDPTABLE Module
            .IDENT  \01-01\

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

 45  4C  42  41  54  50  44  52  43  09  00000 P.AAA:  .ASCII  <9>\CRDPTABLE\          ;

      $MODULE=        P.AAA
       .EXTRN  CRD$FIND_PTABLE
       .EXTRN  CRD$GET_DEVICE_NAME
       .EXTRN  CRD$PRINT_PTABLES
       .EXTRN  CRD$GA_PTABLE

       .PSECT  CODE,NOWRT,  SHR,  PIC,2

         07FC 00000     .ENTRY  CRD$FIND_PTABLE, Save R2,R3,R4,R5,R6,R7,R8,-; 0097
        R9,R10          ;
       5A 00000000G  EF      04  C1 00002    ADDL3   #4, CRD$GA_PTABLE, R10        ; 0124
       55 04   AC  D0 0000A    MOVL    DEVICE_NAME, R5          ; 0137
       65  95 0000E    TSTB    (R5)            ;
       38  13 00010    BEQL    6$            ;
       54 00000000G  FF      01  C3 00012    SUBL3   #1, @CRD$GA_PTABLE, PTABLE_TABLE_INDEX      ; 0143
       28  11 0001A    BRB     5$            ; 0188
       56      58  D0 0001C 1$:   MOVL    PTABLE_PTR, PTABLE_DEVICE_NAME_DESC       ; 0161
       59      66  D0 0001F    MOVL    (PTABLE_DEVICE_NAME_DESC), -         ; 0166
        PTABLE_DEVICE_NAME_LEN          ;
       57 04   A6  D0 00022    MOVL    4(PTABLE_DEVICE_NAME_DESC), -        ; 0167
        PTABLE_DEVICE_NAME          ;
       04  11 00026    BRB     3$            ; 0173
       57  D6 00028 2$:   INCL    PTABLE_DEVICE_NAME          ; 0175
       59  D7 0002A    DECL    PTABLE_DEVICE_NAME_LEN          ; 0176
```

```
      5F    8F      67  91 0002C 3$:   CMPB    (PTABLE_DEVICE_NAME), #95        ; 0173
            F6  13 00030    BEQL    2$              ;
            50      65  9A 00032    MOVZBL  (R5), R0            ; 0188
    50          00      67      59  2D 00035    CMPC5   PTABLE_DEVICE_NAME_LEN, -       ;
      01    A5      0003A    (PTABLE_DEVICE_NAME), #0, R0, 1(R5)       ;
            04  12 0003C    BNEQ    4$              ;
            50      58  D0 0003E    MOVL    PTABLE_PTR, R0          ; 0190
        04 00041    RET             ;
            54  D7 00042 4$:   DECL    PTABLE_TABLE_INDEX          ; 0192
            58    6A44  D0 00044 5$:   MOVL    (R10)[PTABLE_TABLE_INDEX], PTABLE_PTR       ; 0152
            D2  12 00048    BNEQ    1$              ;
            50  D4 0004A 6$:   CLRL    R0              ; 0197
        04 0004C    RET             ;

; Routine Size:  77 bytes,    Routine Base:  CODE + 0000
```

```
;    0198  1 %SBTTL 'CRD$Get_Device_Name Routine'
;    0199  1
;    0200  1 GLOBAL ROUTINE CRD$Get_Device_Name (PTable_Ptr) =        !(01)
;    0201  1
;    0202  1 !++
;    0203  1 ! Functional Description:
;    0204  1 !
;    0205  1 ! Construct an ASCIC string that contains the device name found in the PTable pointed
;    0206  1 ! to by PTable_Ptr.
;    0207  1 !
;    0208  1 ! Formal Input Parameters:
;    0209  1 !
;    0210  1 ! PTable_Ptr : A pointer to a PTable.
;    0211  1 !
;    0212  1 ! Formal Output Parameters:
;    0213  1 !
;    0214  1 ! None
;    0215  1 !
;    0216  1 ! Side Effects:
;    0217  1 !
;    0218  1 ! None
;    0219  1 !
;    0220  1 ! Completion Codes:
;    0221  1 !
;    0222  1 ! A pointer to the device name ASCIC string.
;    0223  1 !--
```

```
;    0224  2 BEGIN     ! Routine CRD$Get_Device_Name
;    0225  2  OWN
;    0226  2   Device_Name : VECTOR [12, BYTE];
;    0227  2
;    0228  2  LOCAL
;    0229  2   PTable_Device_Name_Desc,
;    0230  2   Device_Name_Len,
;    0231  2   Device_Name_Ptr;
;    0232  2
;    0233  2  BIND
;    0234  2   PTable = (.PTable_Ptr) : $DS_HPO_DECL ();
;    0235  2
;    0236  2  PTable_Device_Name_Desc = PTable [HP$Q_Device];
;    0237  2
;    0238  3   BEGIN
;    0239  3   MAP
;    0240  3    PTable_Device_Name_Desc : REF VECTOR [2, LONG];
;    0241  3
;    0242  3   Device_Name_Len = .PTable_Device_Name_Desc [0];
;    0243  3   Device_Name_Ptr = .PTable_Device_Name_Desc [1];
;    0244  3
;    0245  4    BEGIN
;    0246  4    MAP
;    0247  4     Device_Name_Ptr : REF VECTOR [, BYTE];
;    0248  4
;    0249  4    WHILE (.Device_Name_Ptr [0] EQL %C'_') DO
;    0250  5     BEGIN
;    0251  5     Device_Name_Ptr = Device_Name_Ptr [1];
;    0252  5     Device_Name_Len = .Device_Name_Len - 1;
;    0253  4     END;
;    0254  3    END;
;    0255  3
;    0256  3   Device_Name [0] = .Device_Name_Len;
;    0257  3
;    0258  3   CH$MOVE (
;    0259  3    .Device_Name_Len,
;    0260  3    .Device_Name_Ptr,
;    0261  3    Device_Name [1]
;    0262  3   );
;    0263  2   END;
;    0264  2
;    0265  2  RETURN (Device_Name);  ! Return the address of the ASCIC string.
;    0266  1 END;     ! Routine CRD$Get Device_Name      ![01]


         .PSECT  WORK,NOEXE,2

        00000 DEVICE_NAME:
        .BLKB   12


         .PSECT  CODE,NOWRT,  SHR,  PIC,2
```

```
                003C 00000     .ENTRY  CRD$GET_DEVICE_NAME, Save R2,R3,R4,R5       ; 0200
            50 04   AC  D0 00002     MOVL    PTABLE_PTR, PTABLE_DEVICE_NAME_DESC       ; 0236
            51      60  D0 00006     MOVL    (PTABLE_DEVICE_NAME_DESC), DEVICE_NAME_LEN  ; 0242
            50 04   A0  D0 00009     MOVL    4(PTABLE_DEVICE_NAME_DESC), DEVICE_NAME_PTR ; 0243
            04  11 0000D     BRB     2$              ; 0249
            50  D6 0000F 1$:    INCL    DEVICE_NAME_PTR         ; 0251
            51  D7 00011     DECL    DEVICE_NAME_LEN         ; 0252
         5F 8F      60  91 00013 2$:    CMPB    (DEVICE_NAME_PTR), #95          ; 0249
            F6  13 00017     BEQL    1$              ;
          00000000'  EF      51  90 00019     MOVB    DEVICE_NAME_LEN, DEVICE_NAME          ; 0256
          00000000'  EF        60      51  28 00020     MOVC3   DEVICE_NAME_LEN, (DEVICE_NAME_PTR), -       ; 0261
               DEVICE_NAME+1          ;
            50 00000000'  EF  9E 00028     MOVAB   DEVICE_NAME, R0         ; 0265
           04 0002F     RET             ; 0266

; Routine Size:  48 bytes,    Routine Base:  CODE + 004D
```

```
;    0267  1 %SBTTL 'CRD$Print_PTables Routine'
;    0268  1
;    0269  1 GLOBAL ROUTINE CRD$Print_PTables : NOVALUE =
;    0270  1
;    0271  1 !++
;    0272  1 ! Functional Description:
;    0273  1 !
;    0274  1 ! Print the information in the PTables.
;    0275  1 !
;    0276  1 ! Formal Input Parameters:
;    0277  1 !
;    0278  1 ! None
;    0279  1 !
;    0280  1 ! Formal Output Parameters:
;    0281  1 !
;    0282  1 ! None
;    0283  1 !
;    0284  1 ! Side Effects:
;    0285  1 !
;    0286  1 ! None
;    0287  1 !
;    0288  1 ! Completion Codes:
;    0289  1 !
;    0290  1 ! None
;    0291  1 !
```

```
;    0292  2 BEGIN     ! Routine CRD$Print_PTables
;    0293  2  BIND
;    0294  2   Numb_PTable_Addresses = (.CRD$GA_PTable),
;    0295  2   PTable_Addresses      = (.CRD$GA_PTable + 4) : VECTOR [, LONG];
;    0296  2
;    0297  2  LOCAL
;    0298  2   PTable_Index,  ! Index into the PTable_Addresses table
;    0299  2   PTable_Ptr;  ! A pointer to a ptable
;    0300  2
;    0301  2  PTable_Index = .Numb_PTable_Addresses - 1;
;    0302  2      ! Start at the last PTable address
;    0303  2
;    0304  2  WHILE ((PTable_Ptr = .PTable_Addresses [.PTable_Index]) NEQ 0) DO
;    0305  3   BEGIN
;    0306  3   BIND
;    0307  3    PTable = (.PTable_Ptr) : $DS_HPO_DECL (),
;    0308  3
;    0309  3    Output_0 = $ASCIC ('The !UL PTable is:!/'),
;    0310  3    Output_1 = $ASCIC ('** Device Name: .!AS.!/'),
;    0311  3    Output_2 = $ASCIC ('** Address:     !XL(X)!/'),
;    0312  3    Output_3 = $ASCIC ('** Size:        !UW!/'),
;    0313  3    Output_4 = $ASCIC ('** Flags:       !UB!/'),
;    0314  3    Output_5 = $ASCIC ('** Drive:       !UB!/'),
;    0315  3    Output_6 = $ASCIC ('** Link:        !XL(X)!/'),
;    0316  3    Output_7 = $ASCIC ('** Type:        .!AC.!/'),
;    0317  3    Output_8 = $ASCIC ('!/');
;    0318  3
;    0319  3   $CRD_Print (Output_0, .PTable_Index);
;    0320  3   $CRD_Print (Output_1, PTable [HP$Q_Device]);
;    0321  3   $CRD_Print (Output_2, .PTable_Ptr);
;    0322  3   $CRD_Print (Output_3, .PTable [HP$W_Size]);
;    0323  3   $CRD_Print (Output_4, .PTable [HP$B_Refc]);
;    0324  3   $CRD_Print (Output_5, .PTable [HP$B_Drive]);
;    0325  3   $CRD_Print (Output_6, .PTable [HP$A_Link]);
;    0326  3   $CRD_Print (Output_7, PTable [HP$T_Type]);
;    0327  3   $CRD_Print (Output_8);
;    0328  3
;    0329  3   PTable_Index = .PTable_Index - 1;
;    0330  3      ! Go to the next ptable and print it
;    0331  2   END;
;    0332  1 END;     ! Routine CRD$Print_PTables


            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

65  6C  62  61  54  50  20  4C  55  21  20  65  68  54  14  0000A P.AAB:  .ASCII  <20>\The !UL PTable is:!/\        ;
        2F  21  3A  73  69  20  00019                   ;
65  6D  61  4E  20  65  63  69  76  65  44  20  2A  2A  17  0001F P.AAC:  .ASCII  <23>\** Device Name: .!AS.!/\       ;
    2F  21  2E  53  41  21  2E  20  3A  0002E                   ;
20  20  20  3A  73  73  65  72  64  64  41  20  2A  2A  18  00037 P.AAD:  .ASCII  <24>\** Address:     !XL(X)!/\       ;
      2F  21  29  58  28  4C  58  21  20  20  00046                   ;
20  20  20  20  20  20  3A  65  7A  69  53  20  2A  2A  15  00050 P.AAE:  .ASCII  <21>\** Size:        !UW!/\        ;
    2F  21  57  55  21  20  20  0005F                   ;
20  20  20  20  20  3A  73  67  61  6C  46  20  2A  2A  15  00066 P.AAF:  .ASCII  <21>\** Flags:       !UB!/\        ;
```

```
    2F  21  42  55  21  20  20  00075                ;
20  20  20  20  20  3A  65  76  69  72  44  20  2A  2A  15  0007C P.AAG:  .ASCII  <21>\** Drive:       !UB!/\        ;
    2F  21  42  55  21  20  20  0008B                ;
20  20  20  20  20  20  3A  6B  6E  69  4C  20  2A  2A  18  00092 P.AAH:  .ASCII  <24>\** Link:        !XL(X)!/\       ;
      2F  21  29  58  28  4C  58  21  20  20  000A1              ;
20  20  20  20  20  20  3A  65  70  79  54  20  2A  2A  17  000AB P.AAI:  .ASCII  <23>\** Type:        .!AC.!/\        ;
   2F  21  2E  43  41  21  2E  20  20  000BA                   ;
     2F  21  02  000C3 P.AAJ:  .ASCII  <2>\!/\                 ;

        OUTPUT_0=       P.AAB
        OUTPUT_1=       P.AAC
        OUTPUT_2=       P.AAD
        OUTPUT_3=       P.AAE
        OUTPUT_4=       P.AAF
        OUTPUT_5=       P.AAG
        OUTPUT_6=       P.AAH
        OUTPUT_7=       P.AAI
        OUTPUT_8=       P.AAJ
         .EXTRN  CRD$PRINT

         .PSECT  CODE,NOWRT,  SHR,  PIC,2

           001C 00000     .ENTRY  CRD$PRINT_PTABLES, Save R2,R3,R4        ; 0269
         54 00000000G  EF      04  C1 00002    ADDL3   #4, CRD$GA_PTABLE, R4          ; 0295
         53 00000000G  FF      01  C3 0000A    SUBL3   #1, @CRD$GA_PTABLE, PTABLE_INDEX        ; 0301
       00B3  31 00012    BRW     2$            ; 0304
         53  DD 00015 1$:   PUSHL   PTABLE_INDEX           ; 0319
    00000000'  EF  9F 00017    PUSHAB  OUTPUT_0             ;
         01  DD 0001D    PUSHL   #1                ;
         1A  DD 0001F    PUSHL   #26               ;
    00000000G  FF      04  FB 00021    CALLS   #4, @CRD$PRINT          ;
         52  DD 00028    PUSHL   PTABLE_PTR            ; 0320
    00000000'  EF  9F 0002A    PUSHAB  OUTPUT_1             ;
         01  DD 00030    PUSHL   #1                ;
         1A  DD 00032    PUSHL   #26               ;
    00000000G  FF      04  FB 00034    CALLS   #4, @CRD$PRINT          ;
         52  DD 0003B    PUSHL   PTABLE_PTR            ; 0321
    00000000'  EF  9F 0003D    PUSHAB  OUTPUT_2             ;
         01  DD 00043    PUSHL   #1                ;
         1A  DD 00045    PUSHL   #26               ;
    00000000G  FF      04  FB 00047    CALLS   #4, @CRD$PRINT          ;
         7E 08   A2  3C 0004E    MOVZWL  8(PTABLE_PTR), -(SP)          ; 0322
    00000000'  EF  9F 00052    PUSHAB  OUTPUT_3             ;
         01  DD 00058    PUSHL   #1                ;
         1A  DD 0005A    PUSHL   #26               ;
    00000000G  FF      04  FB 0005C    CALLS   #4, @CRD$PRINT          ;
         7E 0A   A2  9A 00063    MOVZBL  10(PTABLE_PTR), -(SP)         ; 0323
    00000000'  EF  9F 00067    PUSHAB  OUTPUT_4             ;
         01  DD 0006D    PUSHL   #1                ;
         1A  DD 0006F    PUSHL   #26               ;
    00000000G  FF      04  FB 00071    CALLS   #4, @CRD$PRINT          ;
         7E 0B   A2  9A 00078    MOVZBL  11(PTABLE_PTR), -(SP)         ; 0324
    00000000'  EF  9F 0007C    PUSHAB  OUTPUT_5             ;
         01  DD 00082    PUSHL   #1                ;
         1A  DD 00084    PUSHL   #26               ;
```

```
                00000000G  FF       04  FB 00086    CALLS   #4, aCRD$PRINT          ;
               20   A2  DD 0008D     PUSHL   32(PTABLE_PTR)          ; 0325
                00000000'  EF  9F 00090    PUSHAB  OUTPUT_6           ;
                     01  DD 00096     PUSHL   #1              ;
                     1A  DD 00098     PUSHL   #26             ;
                00000000G  FF       04  FB 0009A    CALLS   #4, aCRD$PRINT          ;
               26   A2  9F 000A1     PUSHAB  38(PTABLE_PTR)          ; 0326
                00000000'  EF  9F 000A4    PUSHAB  OUTPUT_7           ;
                     01  DD 000AA     PUSHL   #1              ;
                     1A  DD 000AC     PUSHL   #26             ;
                00000000G  FF       04  FB 000AE    CALLS   #4, aCRD$PRINT          ;
                00000000'  EF  9F 000B5    PUSHAB  OUTPUT_8           ; 0327
                     01  DD 000BB     PUSHL   #1              ;
                     1A  DD 000BD     PUSHL   #26             ;
                00000000G  FF       03  FB 000BF    CALLS   #3, aCRD$PRINT          ;
                     53  D7 000C6     DECL    PTABLE_INDEX            ; 0329
                     52    6443  D0 000C8 2$:   MOVL    (R4)[PTABLE_INDEX], PTABLE_PTR      ; 0304
                     03  13 000CC     BEQL    3$              ;
                   FF44  31 000CE     BRW     1$              ;
                    04 000D1 3$:   RET                 ; 0332

; Routine Size:  210 bytes,    Routine Base:  CODE + 007D
```

```
;    0333  1 END     ! End of CRDPTABLE module
;    0334  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          198  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          335  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  WORK           12  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$ASCIC    Macro    Lib03   309     310     311     312     313     314     315     316     317
$CRD_LITERALS   Macro    Lib01    78
$CRD_PRINT   Macro    Lib01   319     320     321     322     323     324     325     326     327
$DS_HPO_DECL   Macro    Lib03   155     234     307
$DS_HPO_F   Fieldset Lib03   155     234     307
$DS_TYPEDEF   Macro    Lib03   319     320     321     322     323     324     325     326     327
$EQULST    Macro    Lib04    78     319     320     321     322     323     324     325     326     327
$ER    Comptime    94
$MODULE    Bind        94
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     78
AUTO$K_EXIT_CONTROL_C  Literal     78
AUTO$K_EXIT_DUMMY_2  Literal     78
AUTO$K_EXIT_DUMMY_3  Literal     78
AUTO$K_EXIT_ERRORS_COMPLETE Literal     78
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     78
AUTO$K_EXIT_INTERNAL_ERROR Literal     78
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     78
AUTO$K_EXIT_LAST_REASON  Literal     78    78
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     78
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     78
AUTO$K_EXIT_NOERRORS_PREP Literal     78
CRD$FIND_PTABLE   GlobRout    97 Lib01e     68f
CRD$GA_PTABLE   External Lib01   123.    124.    294.    295.
CRD$GET_DEVICE_NAME  GlobRout   200 Lib01e     71f
CRD$K_ACTION_EQL_DO_NOTHING Literal     78
CRD$K_ACTION_RANGE_ERROR Literal     78
CRD$K_ADVANCE_DIAGNOSTIC Literal     78
CRD$K_ADVANCE_GENERAL_COM Literal     78
CRD$K_ADVANCE_STATE  Literal     78
CRD$K_ALLOCATION_ERROR  Literal     78
CRD$K_ASSIGN_PTABLE_PTRS Literal     78
CRD$K_AUTOSIZER_ERROR  Literal     78
CRD$K_AUTOSIZER_SET_FLAGS Literal     78
```

Symbol	Type	Defined	Referenced ...
CRD$K_BAD_ACTION_NUMBER	Literal		78
CRD$K_BAD_TYPECODE_ERROR	Literal		78
CRD$K_BASE_SYSTEM_IDC	Literal		78
CRD$K_BASE_SYSTEM_LESI	Literal		78
CRD$K_BASE_SYSTEM_NULL	Literal		78
CRD$K_BASE_SYSTEM_UDA_0	Literal		78
CRD$K_BASE_SYSTEM_UDA_1	Literal		78
CRD$K_BASE_SYSTEM_UDA_2	Literal		78
CRD$K_BASE_SYSTEM_UDA_3	Literal		78
CRD$K_CRD_INITIALIZATION	Literal		78
CRD$K_CREATE_HST	Literal		78
CRD$K_DATA_LENGTH_ERROR	Literal		78
CRD$K_DDB_AUTO_SUPPORT_ENTRY	Literal		78
CRD$K_DDB_BLINK	Literal		78
CRD$K_DDB_FLINK	Literal		78
CRD$K_DDB_LAST_ENTRY	Literal		78
CRD$K_DDB_MEMORY_SIZE	Literal		78
CRD$K_DDB_MENU_DEVICE_NUMB	Literal		78
CRD$K_DDB_MENU_SUPPORT_ENTRY	Literal		78
CRD$K_DDB_MENU_SUPPORT_SIZE	Literal		78
CRD$K_DDB_NUMB_SUPPORT_TABLES	Literal		78
CRD$K_DDB_PTABLE_ENTRY	Literal		78
CRD$K_DDB_STATUS	Literal		78
CRD$K_DDB_SUPBLK_BLOCK_PTR	Literal		78
CRD$K_DEVICE_NAME	Literal		78
CRD$K_DIAGNOSTIC_ERROR	Literal		78
CRD$K_DIAGNOSTIC_NAME	Literal		78
CRD$K_DONE_GENERAL_COMS	Literal		78
CRD$K_DO_NOTHING	Literal		78
CRD$K_DUMMY_10	Literal		78
CRD$K_DUMMY_11	Literal		78
CRD$K_DUMMY_12	Literal		78
CRD$K_DUMMY_13	Literal		78
CRD$K_DUMMY_3	Literal		78
CRD$K_DUMMY_4	Literal		78
CRD$K_DUMMY_5	Literal		78
CRD$K_DUMMY_6	Literal		78
CRD$K_DUMMY_7	Literal		78
CRD$K_DUMMY_8	Literal		78
CRD$K_DUMMY_9	Literal		78
CRD$K_EXIT_CRD	Literal		78
CRD$K_HOOK_POINT	Literal		78
CRD$K_HS_INTERNAL_ERROR	Literal		78
CRD$K_IDC_EVDLB	Literal		78
CRD$K_IDC_EVDLC	Literal		78
CRD$K_IDC_EVDLD	Literal		78
CRD$K_IDC_EVRAD_0	Literal		78
CRD$K_IDC_EVRAD_1	Literal		78
CRD$K_IDC_LAST_DIAGNOSTIC	Literal		78
CRD$K_INTERNAL_ERROR	Literal		78
CRD$K_LAST_ACTION_ROUTINE	Literal		78
CRD$K_LAST_ENTRY	Literal		78
CRD$K_LAST_FUNCTION	Literal		78

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_LAST_HOOK_POINT  Literal      78
CRD$K_LAST_INTERNAL_ERROR Literal      78
CRD$K_LAST_TYPE   Literal      78
CRD$K_LESI_ENWCA  Literal      78
CRD$K_LESI_EVRMB_0  Literal      78
CRD$K_LESI_EVRMB_1  Literal      78
CRD$K_LESI_LAST_DIAGNOSTIC Literal      78
CRD$K_LEVEL_4_PASSED  Literal      78
CRD$K_LOAD_AUTOSIZER  Literal      78
CRD$K_LOAD_ERROR  Literal      78
CRD$K_LOAD_GENERAL_COM  Literal      78
CRD$K_LOAD_LOAD_COM  Literal      78
CRD$K_LOAD_SELECT_COM  Literal      78
CRD$K_LOAD_SET_EVENT_COM Literal      78
CRD$K_LOAD_SET_FLAGS_COM Literal      78
CRD$K_LOAD_START_COM  Literal      78
CRD$K_LOAD_SUP_FILE  Literal      78
CRD$K_MM_INTERNAL_ERROR  Literal      78
CRD$K_NEW_LINE   Literal      78
CRD$K_PREPARATION_ERROR  Literal      78
CRD$K_PREPARATION_ERROR_TEXT Literal      78
CRD$K_RUNTIME   Literal      78
CRD$K_SAME_LINE   Literal      78
CRD$K_SET_EVENT_ARGS  Literal      78
CRD$K_SET_FLAGS_ARGS  Literal      78
CRD$K_SET_UP_DIAGNOSTIC  Literal      78
CRD$K_START   Literal      78
CRD$K_START_AUTOSIZER  Literal      78
CRD$K_START_ERROR  Literal      78
CRD$K_START_QUALIFIERS  Literal      78
CRD$K_STAR_LINE   Literal      78
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal      78
CRD$K_STATE_ADVANCE_GENERAL_COM Literal      78
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal      78
CRD$K_STATE_AUTOSIZER_ERROR Literal      78
CRD$K_STATE_AUTOSIZER_RUNNING Literal      78
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal      78
CRD$K_STATE_DIAGNOSTIC_ERROR Literal      78
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal      78
CRD$K_STATE_DONE_DIAGNOSTICS Literal      78
CRD$K_STATE_DONE_GENERAL_COMS Literal      78
CRD$K_STATE_DUMMY_1  Literal      78
CRD$K_STATE_DUMMY_10  Literal      78
CRD$K_STATE_DUMMY_12  Literal      78
CRD$K_STATE_DUMMY_13  Literal      78
CRD$K_STATE_DUMMY_2  Literal      78
CRD$K_STATE_DUMMY_3  Literal      78
CRD$K_STATE_DUMMY_4  Literal      78
CRD$K_STATE_DUMMY_6  Literal      78
CRD$K_STATE_DUMMY_7  Literal      78
CRD$K_STATE_DUMMY_8  Literal      78
CRD$K_STATE_EXIT_CRD  Literal      78
CRD$K_STATE_INTERNAL_ERROR Literal      78
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_LAST_STATE  Literal     78
CRD$K_STATE_LEVEL_4_PASSED Literal      78
CRD$K_STATE_LOAD_AUTOSIZER Literal      78
CRD$K_STATE_LOAD_ERROR  Literal     78
CRD$K_STATE_LOAD_GENERAL_COM Literal      78
CRD$K_STATE_LOAD_LOAD_COM Literal     78
CRD$K_STATE_LOAD_SELECT_COM Literal      78
CRD$K_STATE_LOAD_SET_EVENT_COM Literal      78
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal      78
CRD$K_STATE_LOAD_START_COM Literal      78
CRD$K_STATE_PREPARATION_ERROR Literal      78
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal      78
CRD$K_STATE_START  Literal     78
CRD$K_STATE_START_AUTOSIZER Literal      78
CRD$K_STATE_START_ERROR  Literal     78
CRD$K_TEST_START_TEXT  Literal     78
CRD$K_TQ_INTERNAL_ERROR  Literal     78
CRD$K_TYPECODE   Literal     78
CRD$K_UDA_0_EVDLB  Literal     78
CRD$K_UDA_0_EVDLC  Literal     78
CRD$K_UDA_0_EVDLD  Literal     78
CRD$K_UDA_0_EVRLA_0  Literal     78
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     78
CRD$K_UDA_1_EVDLB  Literal     78
CRD$K_UDA_1_EVDLC  Literal     78
CRD$K_UDA_1_EVDLD  Literal     78
CRD$K_UDA_1_EVRLA_0  Literal     78
CRD$K_UDA_1_EVRLA_1  Literal     78
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     78
CRD$K_UDA_2_EVDLB  Literal     78
CRD$K_UDA_2_EVDLC  Literal     78
CRD$K_UDA_2_EVDLD  Literal     78
CRD$K_UDA_2_EVRLA_0  Literal     78
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     78
CRD$K_UDA_3_EVDLB  Literal     78
CRD$K_UDA_3_EVDLC  Literal     78
CRD$K_UDA_3_EVDLD  Literal     78
CRD$K_UDA_3_EVRLA_0  Literal     78
CRD$K_UDA_3_EVRLA_1  Literal     78
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     78
CRD$PRINT   External Lib01   319ac   320ac   321ac   322ac   323ac   324ac   325ac   326ac   327ac
CRD$PRINT_PTABLES  GlobRout   269 Lib01e     74f
DEVICE_NAME   RoutForm    97   134m    137a.   188a.
    Own         226   256-    261-    265
DEVICE_NAME_LEN   Local      230   242-    252-    252.    256.    259.
DEVICE_NAME_PTR   Local      231   243-    247m    249a.   251=    251aa   260a.
DS$K_PRINTB   Literal    319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     326
    Literal     327
DS$K_PRINTF     Literal     319   319
    Literal     320   320
    Literal     321   321
    Literal     322   322
    Literal     323   323
    Literal     324   324
    Literal     325   325
    Literal     326   326
    Literal     327   327
DS$K_PRINTI     Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_PRINTX     Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ABORT_PROGRAM  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ABORT_TEST  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_COMMAND_ERR  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_COMMAND_OUT  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_CRD_AUTOTEST  Literal     319   319
    Literal     320   320
    Literal     321   321
    Literal     322   322
    Literal     323   323
    Literal     324   324
    Literal     325   325
    Literal     326   326
    Literal     327   327
DS$K_TYPE_DS_PROMPT  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_DS_START  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ERRDEV  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ERRHARD  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ERROR_BODY  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ERROR_END  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ERRPREP  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ERRSOFT  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ERRSUP  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ERRSYS  Literal     319
    Literal     320
    Literal     321
    Literal     322
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_ERR_HALT  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_EXCEPTION  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_EXCEPTION_HEAD Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_FIRST_PASS  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_GENERAL  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_GENERAL_ERROR  Literal     319
    Literal     320
    Literal     321
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_NO_TESTS  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_PARAM_ERROR  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_PROGRAM_END  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_PROGRAM_INFO  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_PROGRAM_START  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_QIO_INVADP  Litera     319
    Literal     320
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_QIO_NODRIVER  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_QIO_WRONGVER  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_SCRIPT_ECHO  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_SCRIPT_PNF  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_SCRIPT_PROMPT  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_SCRIPT_SKIP  Literal     319
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_SEQUENCE_ERROR Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_START_ERR  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_START_LIST  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_SUMMARY  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
DS$K_TYPE_USER_PROMPT  Literal     319
    Literal     320
    Literal     321
    Literal     322
    Literal     323
    Literal     324
    Literal     325
    Literal     326
    Literal     327
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
GET1ST_     Macro    Lib04    78     319     320     321     322     323     324     325     326     327
GET2ND_     Unbound           78     319     320     321     322     323     324     325     326     327
HP$A_LINK    Field    Lib03   325.
HP$B_DRIVE    Field    Lib03   324.
HP$B_REFC    Field    Lib03   323.
HP$K_LENGTH    Literal  Lib03   155     234     307
HP$Q_DEVICE    Field    Lib03   161a    236a    320a
HP$T_TYPE    Field    Lib03   326a
HP$W_SIZE    Field    Lib03   322.
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal     78
HS$K_ACTION_ADVANCE_GENERAL_COM Literal     78
HS$K_ACTION_ADVANCE_STATE Literal     78
HS$K_ACTION_CHANGE_TABLE Literal     78
HS$K_ACTION_DIAGNOSTIC_ERROR Literal     78
HS$K_ACTION_DONE_GENERAL_COMS Literal     78
HS$K_ACTION_DO_NOTHING  Literal     78
HS$K_ACTION_DUMMY_3  Literal     78
HS$K_ACTION_DUMMY_4  Literal     78
HS$K_ACTION_DUMMY_5  Literal     78
HS$K_ACTION_DUMMY_6  Literal     78
HS$K_ACTION_INIT_HS  Literal     78
HS$K_ACTION_INTERNAL_ERROR Literal     78
HS$K_ACTION_LAST_ACTION  Literal     78
HS$K_ACTION_LOAD_ERROR  Literal     78
HS$K_ACTION_LOAD_GENERAL_COM Literal     78
HS$K_ACTION_LOAD_LOAD_COM Literal     78
HS$K_ACTION_LOAD_SELECT_COM Literal     78
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     78
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     78
HS$K_ACTION_LOAD_START_COM Literal     78
HS$K_ACTION_PREPARATION_ERROR Literal     78
HS$K_ACTION_START_ERROR  Literal     78
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     78
HS$K_STATE_ADVANCE_GENERAL_COM Literal     78
HS$K_STATE_CHANGE_TABLE  Literal     78
HS$K_STATE_DIAGNOSTIC_ERROR Literal     78
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     78
HS$K_STATE_DONE_GENERAL_COMS Literal     78
HS$K_STATE_DUMMY_1  Literal     78
HS$K_STATE_DUMMY_2  Literal     78
HS$K_STATE_DUMMY_3  Literal     78
HS$K_STATE_DUMMY_4  Literal     78
HS$K_STATE_DUMMY_6  Literal     78
HS$K_STATE_INIT_HS  Literal     78
HS$K_STATE_INTERNAL_ERROR Literal     78
HS$K_STATE_LAST_STATE  Literal     78
HS$K_STATE_LOAD_ERROR  Literal     78
HS$K_STATE_LOAD_GENERAL_COM Literal     78
HS$K_STATE_LOAD_LOAD_COM Literal     78
HS$K_STATE_LOAD_SELECT_COM Literal     78
HS$K_STATE_LOAD_SET_EVENT_COM Literal     78
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     78
HS$K_STATE_LOAD_START_COM Literal     78
```

Symbol	Type	Defined	Referenced ...
HS$K_STATE_PREPARATION_ERROR	Literal		78
HS$K_STATE_START_ERROR	Literal		78
MEN$K_HS_STATE_TABLE	Literal		78
MEN$K_MM_STATE_TABLE	Literal		78
MEN$K_TQ_STATE_TABLE	Literal		78
MENU$K_EXIT_AUTOSIZER_ERROR	Literal		78
MENU$K_EXIT_INTERNAL_ERROR	Literal		78
MENU$K_EXIT_LAST_REASON	Literal		78
MENU$K_EXIT_OPERATOR_ABORT	Literal		78
MENU$K_EXIT_OPERATOR_STOP	Literal		78
MENU$K_EXIT_SUP_FILE_LOAD_ERR	Literal		78
MENU$K_EXIT_SUP_FILE_NOT_FOUND	Literal		78
MM$K_ACTION_ADVANCE_STATE	Literal		78
MM$K_ACTION_AUTOSIZER_ERROR	Literal		78
MM$K_ACTION_BUILD_DDBS	Literal		78
MM$K_ACTION_BUILD_HSTS	Literal		78
MM$K_ACTION_CHANGE_TABLE	Literal		78
MM$K_ACTION_DONE_INITS	Literal		78
MM$K_ACTION_DO_NOTHING	Literal		78
MM$K_ACTION_DUMMY_3	Literal		78
MM$K_ACTION_DUMMY_4	Literal		78
MM$K_ACTION_DUMMY_5	Literal		78
MM$K_ACTION_DUMMY_6	Literal		78
MM$K_ACTION_DUMMY_7	Literal		78
MM$K_ACTION_DUMMY_8	Literal		78
MM$K_ACTION_INTERNAL_ERROR	Literal		78
MM$K_ACTION_LAST_ACTION	Literal		78
MM$K_ACTION_LOAD_AUTOSIZER	Literal		78
MM$K_ACTION_MENU_PROMPT	Literal		78
MM$K_ACTION_PRINT_MENU	Literal		78
MM$K_ACTION_PRINT_MENU_HEADING	Literal		78
MM$K_ACTION_SET_FLAGS_AUTOSIZER	Literal		78
MM$K_ACTION_START_AUTOSIZER	Literal		78
MM$K_STATE_AUTOSIZER_ERROR	Literal		78
MM$K_STATE_AUTOSIZER_RUNNING	Literal		78
MM$K_STATE_BUILD_DDBS	Literal		78
MM$K_STATE_BUILD_HSTS	Literal		78
MM$K_STATE_CHANGE_TABLE	Literal		78
MM$K_STATE_DONE_INITS	Literal		78
MM$K_STATE_DUMMY_1	Literal		78
MM$K_STATE_DUMMY_2	Literal		78
MM$K_STATE_DUMMY_3	Literal		78
MM$K_STATE_DUMMY_5	Literal		78
MM$K_STATE_DUMMY_7	Literal		78
MM$K_STATE_DUMMY_8	Literal		78
MM$K_STATE_INTERNAL_ERROR	Literal		78
MM$K_STATE_LAST_STATE	Literal		78
MM$K_STATE_LOAD_AUTOSIZER	Literal		78
MM$K_STATE_MENU_PROMPT	Literal		78
MM$K_STATE_PRINT_MENU	Literal		78
MM$K_STATE_PRINT_MENU_HEADING	Literal		78
MM$K_STATE_SET_FLAGS_AUTOSIZER	Literal		78
MM$K_STATE_START	Literal		78

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
MM$K_STATE_START_AUTOSIZER Literal     78
MODNAM    Macro    Lib02    94
NUL2ND    Macro    Lib04    78     319     320     321     322     323     324     325     326     327
NUMB_PTABLE_ADDRESSES  Bind        294   301.
NUMB_PTABLE_ENTRIES  Bind        123   143.
OUTPUT_0   Bind         309   319a
OUTPUT_1   Bind         310   320a
OUTPUT_2   Bind         311   321a
OUTPUT_3   Bind         312   322a
OUTPUT_4   Bind         313   323a
OUTPUT_5   Bind         314   324a
OUTPUT_6   Bind         315   325a
OUTPUT_7   Bind         316   326a
OUTPUT_8   Bind         317   327a
PTABLE    Bind        155   161a
    Bind        234   236a
    Bind        307   320a    322.    323.    324.    325.    326a
PTABLE_ADDRESSES  Bind        295   304.
PTABLE_DEVICE_NAME  Local       129   167=    171m    173a.   175=    175aa   187a.
PTABLE_DEVICE_NAME_DESC  Local       131   161=    164m    166a.   167a.
    Local       229   236=    240m    242a.   243a.
PTABLE_DEVICE_NAME_LEN  Local       130   166=    176=    176.    187.
PTABLE_INDEX   Local       298   301=    304.    319.    329=    329.
PTABLE_PTR   Local       128   152=    155.    190.
    RoutForm    200   234.
    Local       299   304=    307.    321.
PTABLE_TABLE   Bind        124   152.
PTABLE_TABLE_INDEX  Local       127   143=    152.    192=    192.
TQ$K_ACTION_ADVANCE_STATE Literal     78
TQ$K_ACTION_CHANGE_TABLE Literal     78
TQ$K_ACTION_DONE_TEST_QUEUE Literal     78
TQ$K_ACTION_DO_NOTHING  Literal     78
TQ$K_ACTION_DUMMY_3  Literal     78
TQ$K_ACTION_DUMMY_4  Literal     78
TQ$K_ACTION_DUMMY_5  Literal     78
TQ$K_ACTION_GET_DDB  Literal     78
TQ$K_ACTION_INIT_TQ  Literal     78
TQ$K_ACTION_INTERNAL_ERROR Literal     78
TQ$K_ACTION_LAST_ACTION  Literal     78
TQ$K_STATE_CHANGE_TABLE  Literal     78
TQ$K_STATE_DONE_TEST_QUEUE Literal     78
TQ$K_STATE_DUMMY_1  Literal     78
TQ$K_STATE_DUMMY_2  Literal     78
TQ$K_STATE_DUMMY_3  Literal     78
TQ$K_STATE_DUMMY_4  Literal     78
TQ$K_STATE_DUMMY_5  Literal     78
TQ$K_STATE_GET_DDB  Literal     78
TQ$K_STATE_INIT_TQ  Literal     78
TQ$K_STATE_INTERNAL_ERROR Literal     78
TQ$K_STATE_LAST_STATE  Literal     78
```

```
  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDPTABLE.B32;1
    2  Module CRDPTABLE
   55  Library #1 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   58  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   61  Library #3 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   64  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  334  Eludom CRDPTABLE


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     @ Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;            486          7          1          62          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;            671          1          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;            923         18          1          94          00:00.2
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0       1000          00:04.2




;     COMMAND QUALIFIERS

; BLISS CRDPTABLE.B32/LIST-CRDPTABLE.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDPTABLE.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  335 code + 210 data bytes
; Run Time:    00:35.6
; Elapsed Time:    01:07.4
; Lines/CPU Min:     563
; Lexemes/CPU-Min: 69974
```

```
; Memory Used: 232 pages
; Compilation Complete
```

```
;    0001   0 %TITLE '*** CRDSTATE Module'
;    0002   0 MODULE CRDSTATE (    ! Operate a finite state automata
;    0003   0   IDENT = '01-04'
;    0004   0   ) =
;    0005   0 !++
;    0006   0 ! COPYRIGHT (c) 1980, 1981
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! Operate a finite state automata for the purpose of executing DS
;    0032  0 ! commands for CRD-AutoTest.
;    0033  0 !
;    0034  0 ! Author:
;    0035  0 !
;    0036  0 ! Jack Stansbury
;    0037  0 !
;    0038  0 ! Creation Date:
;    0039  0 !
;    0040  0 ! April 22, 1982
;    0041  0 !
;    0042  0 ! Version:
;    0043  0 !
;    0044  0 ! 6.9
;    0045  0 !
;    0046  0 ! Modified By:
;    0047  0 !
;    0048  0 ! 01 Jack Stansbury, Version 1.1, July 27, 1982
;    0049  0 !  Changed the action routine for the DS_Prompt/Done_General_Coms from
;    0050  0 !  Advance_State to Done_General_Coms!
;    0051  0 !
;    0052  0 ! 02 Jack Stansbury, Version 1.1, September 30, 1982
;    0053  0 !  Major revisions for the changes in state and action routine names. None of the changes
;    0054  0 !  are marked because there were so many.
;    0055  0 !
;    0056  0 ! 03 Jack Stansbury, Version 1.2, March 3, 1983
;    0057  0 !  Since both the error action routines (for diagnostic_error and for preparation_error)
;    0058  0 !  both abort the diagnostic, and since the cleanup section is run as part of the abort,
;    0059  0 !  and since the diagnostic MAY (although not supposed too ...) issue print statements,
;    0060  0 !  error calls, prompts, etc., make the diagnostic_error state and the preparation_error
;    0061  0 !  state both be as forgiving as possible. Only change state out of these two states
;    0062  0 !  when you get the DS> prompt.
;    0063  0 !
;    0064  0 ! 04 Jack Stansbury, Version 1.2, March 4, 1983
;    0065  0 !  Added an Assert statement for the number of typecodes. If the number of typecodes
;    0066  0 !  changes from 38, the state tables must be changed accordingly.
;    0067  0 !--
```

```
;    0068  0 %SBTTL 'Libraries'
;    0069  1 BEGIN     ! Begin the CRDSTATE module
;    0070  1
;    0071  1 LIBRARY
;    0072  1  '$DS';     ! Use Ds.L32 first
;    0073  1
;    0074  1 LIBRARY
;    0075  1  '$Diag';    ! Use Diag.L32 second
;    0076  1
;    0077  1 LIBRARY
;    0078  1  '$CRD';     ! Use CRD.L32 third
;    0079  1
;    0080  1 LIBRARY
;    0081  1  'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0082  1 %SBTTL 'Included Macros and Literals'
;    0083  1
;    0084  1  $DS_TypeDef;    ! Define the TypeCodes
;    0085  1  $CRD_Literals;   ! Define some literals for CRD
;    0086  1
;    0087  1 GLOBAL      ! Define these globally
;    0088  1  $DS_DSSDef;    ! Define Resident-DS entry points for Supervisor services (for $DS_Break)
```

```
;    0089  1 %SBTTL 'Table of Contents'
;    0090  1
;    0091  1 FORWARD ROUTINE
;    0092  1  CRD$Get_State   : ADDRESSING_MODE (LONG_RELATIVE),
;    0093  1       ! Get the current state number
;    0094  1
;    0095  1  CRD$Get_Action_Routine  : ADDRESSING_MODE (LONG_RELATIVE),
;    0096  1       ! Get the current action routine number
;    0097  1
;    0098  1  CRD$Init_State_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0099  1       ! Initialize the CRD$GW_State_Table
;    0100  1
;    0101  1  CRD$Turing_Machine  : ADDRESSING_MODE (LONG_RELATIVE),
;    0102  1       ! Step through the state table
;    0103  1
;    0104  1  CRD$Print_Action_Routine : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
;    0105  1       ! Print the action routine number
```

```
;    0106  1 %SBTTL 'Psect Definitions'
;    0107  1
;    0108  1 PSECT
;    0109  1     PLIT   = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0110  1
;    0111  1 PSECT
;    0112  1     CODE   = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0113  1
;    0114  1 PSECT
;    0115  1     OWN    = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0116  1
;    0117  1 PSECT
;    0118  1     GLOBAL = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0119  1 %SBTTL 'Global Own Storage'
;    0120  1
;    0121  1 GLOBAL
;    0122  1  CRD$GL_Current_TypeCode, ! This is the current typecode number
;    0123  1  CRD$GL_Current_State,  ! This is the current state number
;    0124  1  CRD$GL_Current_Action;  ! This is the current action routine number
```

```
;    0125  1 %SBTTL 'Local Macro Definitions'
;    0126  1
;    0127  1 COMPILETIME
;    0128  1  $Temp$        = 0,  ! For temps
;    0129  1  $Prev_Column$ = 0,  ! The previous column in the Insert_N macro
;    0130  1  $Window_Size$ = 0,  ! The window size in the Insert_N macro
;    0131  1  $Row_Cnt$     = 0,  ! For Fill_N_Rows macro - count rows
;    0132  1  $Entry$       = 0,  ! Number of entrys put into current row
;    0133  1  $Rows$        = 0;  ! Number of rows put in so far
;    0134  1
;    0135  1 MACRO
; M  0136  1  Fill_N_Entrys (N, New_State, Action_Routine) =
; M  0137  1   REP N OF
; M  0138  1    WORD (%NAME ('CRD$K_', Action_Routine),  %NAME ('CRD$K_', New_State))
; M  0139  1
; M  0140  1   %IF ($Entry$ + N) EQL CRD$K_Numb_States
; M  0141  1   %THEN
; M  0142  1    %ASSIGN ($Rows$, $Rows$ + 1)
; M  0143  1    %ASSIGN ($Entry$, 0)
; M  0144  1   %ELSE
; M  0145  1    %IF ($Entry$ + N) GTR CRD$K_Numb_States
; M  0146  1    %THEN
; M  0147  1     %ERRORMACRO ('Fill_N_Entrys: Too many entrys (', %NUMBER ($Entry$), ') in row ', %NUMBER ($Rows$))
; M  0148  1    %ELSE
; M  0149  1     %ASSIGN ($Entry$, $Entry$ + N)
; M  0150  1    %FI
; M  0151  1   %FI
;    0152  1  %;
;    0153  1
;    0154  1 MACRO
; M  0155  1  Insert_Entrys (New_State, Action_Routine) [] =
; M  0156  1   Fill_N_Entrys (1, New_State, Action_Routine)
; M  0157  1
; M  0158  1   %IF NOT %NULL (%REMAINING)
; M  0159  1   %THEN
; M  0160  1    ,Insert_Entrys (%REMAINING)
; M  0161  1   %FI
;    0162  1  %;
;    0163  1
;    0164  1 MACRO
; M  0165  1  Fill_Full_Row =
; M  0166  1   %IF $Entry$ NEQ 0
; M  0167  1   %THEN
; M  0168  1    %ERRORMACRO ('Fill_Full_Row: $Entry$ = ', %NUMBER ($Entry$), ', not zero')
; M  0169  1   %ELSE
; M  0170  1    Fill_N_Entrys (CRD$K_Numb_States, State_Internal_Error, Internal_Error)
; M  0171  1   %FI
;    0172  1  %;
```

```
;    0173  1 MACRO
;    0174  1  !+
;    0175  1  ! Insert entrys into one whole row. The column number tells what column to put 'State' and
;    0176  1  ! 'Action' into. This macro cann be called with any number of parameters as long as the
;    0177  1  ! number is mod 3.
;    0178  1  !-
;    0179  1
; M  0180  1  Insert_N (Column, State, Action) [] =
; M  0181  1   %IF %COUNT EQL 0
; M  0182  1   %THEN
; M  0183  1    %ASSIGN ($Prev_Column$, 0)
; M  0184  1   %FI
; M  0185  1
; M  0186  1   %ASSIGN ($Window_Size$, %NAME ('CRD$K_', Column) - $Prev_Column$)
; M  0187  1
; M  0188  1   %IF $Window_Size$ GTR 0
; M  0189  1   %THEN
; M  0190  1    Fill_N_Entrys ($Window_Size$, State_Internal_Error, Internal_Error),
; M  0191  1   %FI
; M  0192  1
; M  0193  1   Insert_Entrys (State, Action)
; M  0194  1
; M  0195  1   %ASSIGN ($Prev_Column$, %NAME ('CRD$K_', Column) + 1)
; M  0196  1
; M  0197  1   %IF NOT %NULL (%REMAINING)
; M  0198  1   %THEN
; M  0199  1    , Insert_N (%REMAINING)
; M  0200  1   %ELSE
; M  0201  1    %ASSIGN ($Window_Size$, CRD$K_Numb_States - $Prev_Column$)
; M  0202  1
; M  0203  1    %IF $Window_Size$ GTR 0
; M  0204  1    %THEN
; M  0205  1     , Fill_N_Entrys ($Window_Size$, State_Internal_Error, Internal_Error)
; M  0206  1    %FI
; M  0207  1   %FI
;    0208  1  %;
```

```
;    0209  1 %SBTTL 'Global Bindings'
;    0210  1
; L  0211  1 CRD_Assert ((CRD$K_Numb_Typecodes EQL 38), 'You must add more rows to the state tables for the new typecodes');  ![04]
; %PRINT: ASSERT: The assertion is true.
;    0212  1
;    0213  1 GLOBAL BIND
;    0214  1  CRD$GW_State_Table =
;    0215  1   UPLIT (
; P  0216  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Running, Do_Nothing,
; P  0217  1       State_Diagnostic_Running,  State_Diagnostic_Running, Do_Nothing,
; P  0218  1       State_Internal_Error,  State_Start,   Internal_Error,
; P  0219  1       State_Load_Error,   State_Load_Error,   Do_Nothing,
; P  0220  1       State_Start_Error,   State_Start_Error,   Do_Nothing,
; P  0221  1       State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;    0222  1       State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;    0223  1      ! This row is for the General typecode (Row 0)
;    0224  1
; P  0225  1    Insert_N (State_Start,   State_Level_4_Passed,  Advance_State,
; P  0226  1
; P  0227  1       State_Level_4_Passed,  State_Load_AutoSizer,  Level_4_Passed,
; P  0228  1
; P  0229  1       State_Load_AutoSizer,  State_AutoSizer_Set_Flags, Load_AutoSizer,
; P  0230  1       State_AutoSizer_Set_Flags, State_Start_AutoSizer,  AutoSizer_Set_Flags,
; P  0231  1       State_Start_AutoSizer, State_AutoSizer_Running, Start_AutoSizer,
; P  0232  1       State_AutoSizer_Running, State_Assign_PTable_Ptrs, Advance_State,
; P  0233  1       State_Assign_PTable_Ptrs, State_Set_Up_Diagnostic, Assign_PTable_Ptrs,
; P  0234  1
; P  0235  1       State_Set_Up_Diagnostic, State_Load_General_Com,  Set_Up_Diagnostic,
; P  0236  1
; P  0237  1       State_Load_General_Com, State_Advance_General_Com, Load_General_Com,
; P  0238  1       State_Advance_General_Com, State_Load_General_Com,  Advance_General_Com,
; P  0239  1       State_Done_General_Coms, State_Load_Load_Com,  Done_General_Coms,
; P  0240  1
; P  0241  1       State_Load_Load_Com,  State_Load_Set_Flags_Com, Load_Load_Com,
; P  0242  1       State_Load_Set_Flags_Com, State_Load_Set_Event_Com, Load_Set_Flags_Com,
; P  0243  1       State_Load_Set_Event_Com, State_Load_Select_Com,  Load_Set_Event_Com,
; P  0244  1       State_Load_Select_Com, State_Load_Start_Com,  Load_Select_Com,
; P  0245  1       State_Load_Start_Com,  State_Diagnostic_Running, Load_Start_Com,
; P  0246  1       State_Diagnostic_Running, State_Advance_Diagnostic, Advance_State,
; P  0247  1
; P  0248  1       State_Advance_Diagnostic, State_Set_Up_Diagnostic, Advance_Diagnostic,
; P  0249  1       State_Done_Diagnostics, State_Exit_CRD,   Advance_State,
; P  0250  1       State_Exit_CRD,  State_Start,   Exit_CRD,
; P  0251  1
; P  0252  1       State_AutoSizer_Error,  State_Start,   AutoSizer_Error,
; P  0253  1       State_Internal_Error,  State_Start,   Internal_Error,
; P  0254  1       State_Load_Error,  State_Advance_Diagnostic, Load_Error,
; P  0255  1       State_Start_Error,  State_Advance_Diagnostic, Start_Error,
; P  0256  1       State_Diagnostic_Error, State_Advance_Diagnostic, Advance_State,
;    0257  1       State_Preparation_Error, State_Advance_Diagnostic, Advance_State),
;    0258  1      ! This row is for the DS_Prompt typecode (Row 1)
;    0259  1
; P  0260  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
; P  0261  1       State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
; P  0262  1       State Internal_Error,  State_Start,   Internal_Error,
```

```
;P 0263  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0264  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0265  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0266  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0267  1      ! This row is for the User_Prompt typecode (Row 2)
;   0268  1
;P 0269  1    Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;P 0270  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;P 0271  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0272  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;P 0273  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
;P 0274  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;P 0275  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;P 0276  1        State_Diagnostic_Running, State_Start_Error,  Advance_State,
;P 0277  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0278  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0279  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0280  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0281  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0282  1      ! This row is for the General_Error typecode (Row 3)
;   0283  1
;   0284  1    Fill_Full_Row,
;   0285  1      ! This row is for the ErrSup typecode (Row 4)
;   0286  1
;P 0287  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0288  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0289  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0290  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0291  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0292  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0293  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0294  1      ! This row is for the ErrSys typecode (Row 5)
;   0295  1
;P 0296  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0297  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0298  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0299  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0300  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0301  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0302  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0303  1      ! This row is for the ErrHard typecode (Row 6)
;   0304  1
;P 0305  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0306  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0307  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0308  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0309  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0310  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0311  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0312  1      ! This row is for the ErrSoft typecode (Row 7)
;   0313  1
;P 0314  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0315  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0316  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0317  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
```

```
;P 0318  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0319  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;  0320  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;  0321  1      ! This row is for the ErrDev typecode (Row 8)
;  0322  1
;P 0323  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0324  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0325  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0326  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0327  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;  0328  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;  0329  1      ! This row is for the Error_Body typecode (Row 9)
;  0330  1
;P 0331  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0332  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0333  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0334  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0335  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;  0336  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;  0337  1      ! This row is for the Error_End typecode (Row 10)
;  0338  1
;P 0339  1    Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;P 0340  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;P 0341  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0342  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;P 0343  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
;P 0344  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;P 0345  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;P 0346  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;  0347  1        State_Internal_Error,  State_Start,   Internal_Error),
;  0348  1      ! This row is for the Exception_Head typecode (Row 11)
;  0349  1
;P 0350  1    Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;P 0351  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;P 0352  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0353  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;P 0354  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
;P 0355  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;P 0356  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;P 0357  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0358  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0359  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0360  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0361  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;  0362  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;  0363  1      ! This row is for the Exception typecode (Row 12)
;  0364  1
;P 0365  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0366  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0367  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0368  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0369  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0370  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;  0371  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;  0372  1      ! This row is for the Err_Halt typecode (Row 13)
```

```
;    0373  1
; P  0374  1     Insert_N (State_AutoSizer_Running, State_AutoSizer_Running, Do_Nothing,
; P  0375  1        State_Diagnostic_Running, State_Diagnostic_Running, Do_Nothing,
; P  0376  1        State_Internal_Error,  State_Start,   Internal_Error,
; P  0377  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
; P  0378  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
; P  0379  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;    0380  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;    0381  1      ! This row is for the Summary typecode (Row 14)
;    0382  1
; P  0383  1     Insert_N (State_AutoSizer_Running, State_AutoSizer_Running, Do_Nothing,
; P  0384  1        State_Diagnostic_Running, State_Diagnostic_Running, Do_Nothing,
;    0385  1        State_Internal_Error,  State_Start,   Internal_Error),
;    0386  1      ! This row is for the Program_Start typecode (Row 15)
;    0387  1
; P  0388  1     Insert_N (State_AutoSizer_Running, State_AutoSizer_Running, Do_Nothing,
; P  0389  1        State_Diagnostic_Running, State_Diagnostic_Running, Do_Nothing,
; P  0390  1        State_Internal_Error,  State_Start,   Internal_Error,
; P  0391  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
; P  0392  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
; P  0393  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;    0394  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;    0395  1      ! This row is for the Program_End typecode (Row 16)
;    0396  1
; P  0397  1     Insert_N (State_AutoSizer_Running, State_AutoSizer_Running, Do_Nothing,
; P  0398  1        State_Diagnostic_Running, State_Diagnostic_Running, Do_Nothing,
;    0399  1        State_Internal_Error,  State_Start,   Internal_Error),
;    0400  1      ! This row is for the First_Pass typecode (Row 17)
;    0401  1
; P  0402  1     Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
; P  0403  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
; P  0404  1        State_Internal_Error,  State_Start,   Internal_Error,
; P  0405  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
; P  0406  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
; P  0407  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;    0408  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;    0409  1      ! This row is for the No_Tests typecode (Row 18)
;    0410  1
; P  0411  1     Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
; P  0412  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
; P  0413  1        State_Internal_Error,  State_Start,   Internal_Error,
; P  0414  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
; P  0415  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
; P  0416  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;    0417  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;    0418  1      ! This row is for the Abort_Test typecode (Row 19)
;    0419  1
; P  0420  1     Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
; P  0421  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
; P  0422  1        State_AutoSizer_Error, State_AutoSizer_Error,  Do_Nothing,
; P  0423  1        State_Internal_Error,  State_Start,   Internal_Error,
; P  0424  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
; P  0425  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
; P  0426  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,
;    0427  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),
```

```
;    0428  1      ! This row is for the Abort_Program typecode (Row 20)
;    0429  1
; P  0430  1    Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
; P  0431  1       State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
; P  0432  1       State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
; P  0433  1       State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
; P  0434  1       State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
; P  0435  1       State_Load_Select_Com, State_Start_Error,  Advance_State,
; P  0436  1       State_Load_Start_Com,  State_Start_Error,  Advance_State,
; P  0437  1       State_Diagnostic_Running, State_Start_Error,  Advance_State,
; P  0438  1       State_Internal_Error,  State_Start,   Internal_Error,
; P  0439  1       State_Load_Error,  State_Load_Error,  Do_Nothing,
; P  0440  1       State_Start_Error,  State_Start_Error,  Do_Nothing,
; P  0441  1       State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;    0442  1       State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;    0443  1      ! This row is for the Command_Err typecode (Row 21)
;    0444  1
; P  0445  1    Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
; P  0446  1       State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
; P  0447  1       State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
; P  0448  1       State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
; P  0449  1       State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
; P  0450  1       State_Load_Select_Com, State_Start_Error,  Advance_State,
; P  0451  1       State_Load_Start_Com,  State_Start_Error,  Advance_State,
; P  0452  1       State_Diagnostic_Running, State_Start_Error,  Advance_State,
; P  0453  1       State_Internal_Error,  State_Start,   Internal_Error,
; P  0454  1       State_Load_Error,  State_Load_Error,  Do_Nothing,
; P  0455  1       State_Start_Error,  State_Start_Error,  Do_Nothing,
; P  0456  1       State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;    0457  1       State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;    0458  1      ! This row is for the Command_Out typecode (Row 22)
;    0459  1
; P  0460  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Running, Do_Nothing,
; P  0461  1       State_Diagnostic_Running, State_Diagnostic_Running, Do_Nothing,
;    0462  1       State_Internal_Error,  State_Start,   Internal_Error),
;    0463  1      ! This row is for the Program_Info typecode (Row 23)
;    0464  1
; P  0465  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
; P  0466  1       State_Diagnostic_Running, State_Start_Error,  Advance_State,
;    0467  1       State_Internal_Error,  State_Start,   Internal_Error),
;    0468  1      ! This row is for the Start_Err typecode (Row 24)
;    0469  1
; P  0470  1    Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
; P  0471  1       State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
; P  0472  1       State_Internal_Error,  State_Start,   Internal_Error,
; P  0473  1       State_Load_Error,  State_Load_Error,  Do_Nothing,
; P  0474  1       State_Start_Error,  State_Start_Error,  Do_Nothing,
; P  0475  1       State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;    0476  1       State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;    0477  1      ! This row is for the Sequence_Error typecode (Row 25)
;    0478  1
;    0479  1    Fill_Full_Row,
;    0480  1      ! This row is for the CRD_AutoTest typecode (Row 26)
;    0481  1
; P  0482  1    Insert_N (State_AutoSizer Running, State_AutoSizer_Error,  AutoSizer_Error,
```

```
;P 0483  1        State_Diagnostic_Running, State_Preparation_Error, Preparation_Error,
;P 0484  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0485  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0486  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0487  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0488  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0489  1      ! This row is for the ErrPrep typecode (Row 27)
;   0490  1
;P 0491  1     Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0492  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0493  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0494  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0495  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0496  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0497  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0498  1      ! This row is for the Param_Error typecode (Row 28)
;   0499  1
;   0500  1     Insert_N (State_Start,    State_Start,    Do_Nothing),
;   0501  1      ! This row is for the DS_Start typecode (Row 29)
;   0502  1
;P 0503  1     Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;P 0504  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;P 0505  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0506  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;P 0507  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
;P 0508  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;P 0509  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;P 0510  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0511  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0512  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0513  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0514  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0515  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0516  1      ! This row is for the Script_Pnf typecode (Row 30)
;   0517  1
;P 0518  1     Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;P 0519  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;P 0520  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0521  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;P 0522  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
;P 0523  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;P 0524  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;P 0525  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0526  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0527  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0528  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0529  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0530  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0531  1      ! This row is for the Script_Skip typecode (Row 31)
;   0532  1
;P 0533  1     Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;P 0534  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;P 0535  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0536  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;P 0537  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
```

```
;  P 0538  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;  P 0539  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;  P 0540  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;  P 0541  1        State_Internal_Error,  State_Start,   Internal_Error,
;  P 0542  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;  P 0543  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;  P 0544  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;     0545  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;     0546  1      ! This row is for the Script_Prompt typecode (Row 32)
;     0547  1
;  P 0548  1     Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;  P 0549  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;  P 0550  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;  P 0551  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;  P 0552  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
;  P 0553  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;  P 0554  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;  P 0555  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;  P 0556  1        State_Internal_Error,  State_Start,   Internal_Error,
;  P 0557  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;  P 0558  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;  P 0559  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;     0560  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;     0561  1      ! This row is for the Script_Echo typecode (Row 33)
;     0562  1
;  P 0563  1     Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;  P 0564  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;  P 0565  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;  P 0566  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;  P 0567  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
;  P 0568  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;  P 0569  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;  P 0570  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;  P 0571  1        State_Internal_Error,  State_Start,   Internal_Error,
;  P 0572  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;  P 0573  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;  P 0574  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;     0575  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;     0576  1      ! This row is for the QIO_NoDriver typecode (Row 34)
;     0577  1
;  P 0578  1     Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;  P 0579  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;  P 0580  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;  P 0581  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;  P 0582  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
;  P 0583  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;  P 0584  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;  P 0585  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;  P 0586  1        State_Internal_Error,  State_Start,   Internal_Error,
;  P 0587  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;  P 0588  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;  P 0589  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;     0590  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;     0591  1      ! This row is for the QIO_WrongVer typecode (Row 35)
;     0592  1
```

```
;P 0593  1     Insert_N (State_AutoSizer_Set_Flags, State_AutoSizer_Error,  AutoSizer_Error,
;P 0594  1        State_Start_AutoSizer, State_AutoSizer_Error,  AutoSizer_Error,
;P 0595  1        State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0596  1        State_Load_Set_Flags_Com, State_Load_Error,  Advance_State,
;P 0597  1        State_Load_Set_Event_Com, State_Start_Error,  Advance_State,
;P 0598  1        State_Load_Select_Com, State_Start_Error,  Advance_State,
;P 0599  1        State_Load_Start_Com,  State_Start_Error,  Advance_State,
;P 0600  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0601  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0602  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0603  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0604  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0605  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing),  ! [03]
;   0606  1      ! This row is for the QIO_InvAdp typecode (Row 36)
;   0607  1
;P 0608  1     Insert_N (State_AutoSizer_Running, State_AutoSizer_Error,  AutoSizer_Error,
;P 0609  1        State_Diagnostic_Running, State_Diagnostic_Error,  Diagnostic_Error,
;P 0610  1        State_Internal_Error,  State_Start,   Internal_Error,
;P 0611  1        State_Load_Error,  State_Load_Error,  Do_Nothing,
;P 0612  1        State_Start_Error,  State_Start_Error,  Do_Nothing,
;P 0613  1        State_Diagnostic_Error, State_Diagnostic_Error,  Do_Nothing,  ! [03]
;   0614  1        State_Preparation_Error, State_Preparation_Error, Do_Nothing)  ! [03]
;   0615  1      ! This row is for the Start_List typecode (Row 37)
;   0616  1
;   0617  1    ) : State_Table_Structure [CRD$K_Numb_TypeCodes, CRD$K_Numb_States];
```

```
;    0618  1 %SBTTL 'CRD$Init_State_Table Routine'
;    0619  1
;    0620  1 GLOBAL ROUTINE CRD$Init_State_Table : NOVALUE =
;    0621  1
;    0622  1 !++
;    0623  1 ! Functional Description:
;    0624  1 !
;    0625  1 ! Initialize the CRD$GW_State_Table and all its associated data structures.
;    0626  1 !
;    0627  1 ! Formal Input Parameters:
;    0628  1 !
;    0629  1 ! None
;    0630  1 !
;    0631  1 ! Formal Output Parameters:
;    0632  1 !
;    0633  1 ! None
;    0634  1 !
;    0635  1 ! Side Effects:
;    0636  1 !
;    0637  1 ! Plenty!
;    0638  1 !
;    0639  1 ! Completion Codes:
;    0640  1 !
;    0641  1 ! None
;    0642  1   -
```

```
;   0643  2 BEGIN     ! Routine CRD$Init_State_Table
;   0644  2
;   0645  2  CRD$GL_Current_TypeCode = DS$K_Type_General;
;   0646  2  CRD$GL_Current_State    = CRD$K_State_Start;
;   0647  2  CRD$GL_Current_Action   = CRD$K_Last_Action_Routine;
;   0648  2
;   0649  2  RETURN;
;   0650  1 END;     ! Routine CRD$Init_State_Table


          .TITLE  CRDSTATE *** CRDSTATE Module
          .IDENT  \01-04\

          .PSECT  WORK,NOEXE,2

          00000 CRD$GL_CURRENT_TYPECODE::
          .BLKB   4
          00004 CRD$GL_CURRENT_STATE::
          .BLKB   4
          00008 CRD$GL_CURRENT_ACTION::
          .BLKB   4

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

     001E  001E  00000 P.AAA:  .WORD   30, 30          ;
     001E  001E  00004     .WORD   30, 30          ;
     001E  001E  00008     .WORD   30, 30          ;
     001E  001E  0000C     .WORD   30, 30          ;
     001E  001E  00010     .WORD   30, 30          ;
     001E  001E  00014     .WORD   30, 30          ;
     001E  001E  00018     .WORD   30, 30          ;
     001E  001E  0001C     .WORD   30, 30          ;
     001E  001E  00020     .WORD   30, 30          ;
     0009  0000  00024     .WORD   0, 9          ;
     001E  001E  00028     .WORD   30, 30          ;
     001E  001E  0002C     .WORD   30, 30          ;
     001E  001E  00030     .WORD   30, 30          ;
     001E  001E  00034     .WORD   30, 30          ;
     001E  001E  00038     .WORD   30, 30          ;
     001E  001E  0003C     .WORD   30, 30          ;
     001E  001E  00040     .WORD   30, 30          ;
     001E  001E  00044     .WORD   30, 30          ;
     001E  001E  00048     .WORD   30, 30          ;
     001E  001E  0004C     .WORD   30, 30          ;
     001E  001E  00050     .WORD   30, 30          ;
     001E  001E  00054     .WORD   30, 30          ;
     001E  001E  00058     .WORD   30, 30          ;
     0017  0000  0005C     .WORD   0, 23           ;
     001E  001E  00060     .WORD   30, 30          ;
     001E  001E  00064     .WORD   30, 30          ;
     001E  001E  00068     .WORD   30, 30          ;
     001E  001E  0006C     .WORD   30, 30          ;
     001E  001E  00070     .WORD   30, 30          ;
     001E  001E  00074     .WORD   30, 30          ;
     0003  001E  00078     .WORD   30, 3           ;
```

```
001F  0000  0007C     .WORD   0, 31           ;
0020  0000  00080     .WORD   0, 32           ;
0021  0000  00084     .WORD   0, 33           ;
0022  0000  00088     .WORD   0, 34           ;
001E  001E  0008C     .WORD   30, 30          ;
001E  001E  00090     .WORD   30, 30          ;
001E  001E  00094     .WORD   30, 30          ;
001E  001E  00098     .WORD   30, 30          ;
0004  0001  0009C     .WORD   1, 4           ;
0006  0004  000A0     .WORD   4, 6           ;
001E  001E  000A4     .WORD   30, 30          ;
0007  0006  000A8     .WORD   6, 7           ;
0008  0007  000AC     .WORD   7, 8           ;
0009  0008  000B0     .WORD   8, 9           ;
000A  0001  000B4     .WORD   1, 10           ;
000C  000A  000B8     .WORD   10, 12          ;
001E  001E  000BC     .WORD   30, 30          ;
000E  000C  000C0     .WORD   12, 14          ;
001E  001E  000C4     .WORD   30, 30          ;
000F  000E  000C8     .WORD   14, 15          ;
000E  000F  000CC     .WORD   15, 14          ;
0012  0010  000D0     .WORD   16, 18          ;
001E  001E  000D4     .WORD   30, 30          ;
0013  0012  000D8     .WORD   18, 19          ;
0014  0013  000DC     .WORD   19, 20          ;
0015  0014  000E0     .WORD   20, 21          ;
0016  0015  000E4     .WORD   21, 22          ;
0017  0016  000E8     .WORD   22, 23          ;
0019  0001  000EC     .WORD   1, 25           ;
001E  001E  000F0     .WORD   30, 30          ;
000C  0019  000F4     .WORD   25, 12          ;
001B  0001  000F8     .WORD   1, 27           ;
0003  001B  000FC     .WORD   27, 3           ;
001E  001E  00100     .WORD   30, 30          ;
0003  001D  00104     .WORD   29, 3           ;
0003  001E  00108     .WORD   30, 3           ;
0019  001F  0010C     .WORD   31, 25          ;
0019  0020  00110     .WORD   32, 25          ;
0019  0001  00114     .WORD   1, 25           ;
0019  0001  00118     .WORD   1, 25           ;
001E  001E  0011C     .WORD   30, 30          ;
001E  001E  00120     .WORD   30, 30          ;
001E  001E  00124     .WORD   30, 30          ;
001E  001E  00128     .WORD   30, 30          ;
001E  001E  0012C     .WORD   30, 30          ;
001E  001E  00130     .WORD   30, 30          ;
001E  001E  00134     .WORD   30, 30          ;
001E  001E  00138     .WORD   30, 30          ;
001E  001E  0013C     .WORD   30, 30          ;
001E  001E  00140     .WORD   30, 30          ;
001D  001D  00144     .WORD   29, 29          ;
001E  001E  00148     .WORD   30, 30          ;
001E  001E  0014C     .WORD   30, 30          ;
001E  001E  00150     .WORD   30, 30          ;
001E  001E  00154     .WORD   30, 30          ;
```

```
001E  001E  00158     .WORD   30, 30          ;
001E  001E  0015C     .WORD   30, 30          ;
001E  001E  00160     .WORD   30, 30          ;
001E  001E  00164     .WORD   30, 30          ;
001E  001E  00168     .WORD   30, 30          ;
001E  001E  0016C     .WORD   30, 30          ;
001E  001E  00170     .WORD   30, 30          ;
001E  001E  00174     .WORD   30, 30          ;
001E  001E  00178     .WORD   30, 30          ;
0021  0021  0017C     .WORD   33, 33          ;
001E  001E  00180     .WORD   30, 30          ;
001E  001E  00184     .WORD   30, 30          ;
001E  001E  00188     .WORD   30, 30          ;
001E  001E  0018C     .WORD   30, 30          ;
001E  001E  00190     .WORD   30, 30          ;
001E  001E  00194     .WORD   30, 30          ;
0003  001E  00198     .WORD   30, 3           ;
001F  0000  0019C     .WORD   0, 31           ;
0020  0000  001A0     .WORD   0, 32           ;
0021  0000  001A4     .WORD   0, 33           ;
0022  0000  001A8     .WORD   0, 34           ;
001E  001E  001AC     .WORD   30, 30          ;
001E  001E  001B0     .WORD   30, 30          ;
001E  001E  001B4     .WORD   30, 30          ;
001E  001E  001B8     .WORD   30, 30          ;
001E  001E  001BC     .WORD   30, 30          ;
001E  001E  001C0     .WORD   30, 30          ;
001E  001E  001C4     .WORD   30, 30          ;
001E  001E  001C8     .WORD   30, 30          ;
001D  001D  001CC     .WORD   29, 29          ;
001D  001D  001D0     .WORD   29, 29          ;
001D  001D  001D4     .WORD   29, 29          ;
001E  001E  001D8     .WORD   30, 30          ;
001E  001E  001DC     .WORD   30, 30          ;
001E  001E  001E0     .WORD   30, 30          ;
001E  001E  001E4     .WORD   30, 30          ;
001E  001E  001E8     .WORD   30, 30          ;
001E  001E  001EC     .WORD   30, 30          ;
001E  001E  001F0     .WORD   30, 30          ;
001E  001E  001F4     .WORD   30, 30          ;
001E  001E  001F8     .WORD   30, 30          ;
001F  0001  001FC     .WORD   1, 31           ;
0020  0001  00200     .WORD   1, 32           ;
0020  0001  00204     .WORD   1, 32           ;
0020  0001  00208     .WORD   1, 32           ;
0020  0001  0020C     .WORD   1, 32           ;
001E  001E  00210     .WORD   30, 30          ;
001E  001E  00214     .WORD   30, 30          ;
001E  001E  00218     .WORD   30, 30          ;
001E  001E  0021C     .WORD   30, 30          ;
001E  001E  00220     .WORD   30, 30          ;
001E  001E  00224     .WORD   30, 30          ;
0003  001E  00228     .WORD   30, 3           ;
001F  0000  0022C     .WORD   0, 31           ;
0020  0000  00230     .WORD   0, 32           ;
```

```
      0021  0000  00234     .WORD   0, 33           ;
      0022  0000  00238     .WORD   0, 34           ;
      001E  001E  0023C     .WORD   30, 30          ;
      001E  001E  00240     .WORD   30, 30          ;
      001E  001E  00244     .WORD   30, 30          ;
      001E  001E  00248     .WORD   30, 30          ;
      001E  001E  0024C     .WORD   30, 30          ;
      001E  001E  00250     .WORD   30, 30          ;
      001E  001E  00254     .WORD   30, 30          ;
      001E  001E  00258     .WORD   30, 30          ;
      001E  001E  0025C     .WORD   30, 30          ;
      001E  001E  00260     .WORD   30, 30          ;
      001E  001E  00264     .WORD   30, 30          ;
      001E  001E  00268     .WORD   30, 30          ;
      001E  001E  0026C     .WORD   30, 30          ;
      001E  001E  00270     .WORD   30, 30          ;
      001E  001E  00274     .WORD   30, 30          ;
      001E  001E  00278     .WORD   30, 30          ;
      001E  001E  0027C     .WORD   30, 30          ;
      001E  001E  00280     .WORD   30, 30          ;
      001E  001E  00284     .WORD   30, 30          ;
      001E  001E  00288     .WORD   30, 30          ;
      001E  001E  0028C     .WORD   30, 30          ;
      001E  001E  00290     .WORD   30, 30          ;
      001E  001E  00294     .WORD   30, 30          ;
      001E  001E  00298     .WORD   30, 30          ;
      001E  001E  0029C     .WORD   30, 30          ;
      001E  001E  002A0     .WORD   30, 30          ;
      001E  001E  002A4     .WORD   30, 30          ;
      001E  001E  002A8     .WORD   30, 30          ;
      001E  001E  002AC     .WORD   30, 30          ;
      001E  001E  002B0     .WORD   30, 30          ;
      001E  001E  002B4     .WORD   30, 30          ;
      001E  001E  002B8     .WORD   30, 30          ;
      001E  001E  002BC     .WORD   30, 30          ;
      001E  001E  002C0     .WORD   30, 30          ;
      001E  001E  002C4     .WORD   30, 30          ;
      001E  001E  002C8     .WORD   30, 30          ;
      001E  001E  002CC     .WORD   30, 30          ;
      001E  001E  002D0     .WORD   30, 30          ;
      001E  001E  002D4     .WORD   30, 30          ;
      001E  001E  002D8     .WORD   30, 30          ;
      001E  001E  002DC     .WORD   30, 30          ;
      001E  001E  002E0     .WORD   30, 30          ;
      001E  001E  002E4     .WORD   30, 30          ;
      001E  001E  002E8     .WORD   30, 30          ;
      001E  001E  002EC     .WORD   30, 30          ;
      001E  001E  002F0     .WORD   30, 30          ;
      001D  001D  002F4     .WORD   29, 29          ;
      001E  001E  002F8     .WORD   30, 30          ;
      001E  001E  002FC     .WORD   30, 30          ;
      001E  001E  00300     .WORD   30, 30          ;
      001E  001E  00304     .WORD   30, 30          ;
      001E  001E  00308     .WORD   30, 30          ;
      001E  001E  003CC     .WORD   30, 30          ;
```

```
      001E  001E  00310     .WORD   30, 30           ;
      001E  001E  00314     .WORD   30, 30           ;
      001E  001E  00318     .WORD   30, 30           ;
      001E  001E  0031C     .WORD   30, 30           ;
      001E  001E  00320     .WORD   30, 30           ;
      001E  001E  00324     .WORD   30, 30           ;
      001E  001E  00328     .WORD   30, 30           ;
      0021  0021  0032C     .WORD   33, 33           ;
      001E  001E  00330     .WORD   30, 30           ;
      001E  001E  00334     .WORD   30, 30           ;
      001E  001E  00338     .WORD   30, 30           ;
      001E  001E  0033C     .WORD   30, 30           ;
      001E  001E  00340     .WORD   30, 30           ;
      001E  001E  00344     .WORD   30, 30           ;
      0003  001E  00348     .WORD   30, 3            ;
      001F  0000  0034C     .WORD   0, 31            ;
      0020  0000  00350     .WORD   0, 32            ;
      0021  0000  00354     .WORD   0, 33            ;
      0022  0000  00358     .WORD   0, 34            ;
      001E  001E  0035C     .WORD   30, 30           ;
      001E  001E  00360     .WORD   30, 30           ;
      001E  001E  00364     .WORD   30, 30           ;
      001E  001E  00368     .WORD   30, 30           ;
      001E  001E  0036C     .WORD   30, 30           ;
      001E  001E  00370     .WORD   30, 30           ;
      001E  001E  00374     .WORD   30, 30           ;
      001E  001E  00378     .WORD   30, 30           ;
      001E  001E  0037C     .WORD   30, 30           ;
      001E  001E  00380     .WORD   30, 30           ;
      001D  001D  00384     .WORD   29, 29           ;
      001E  001E  00388     .WORD   30, 30           ;
      001E  001E  0038C     .WORD   30, 30           ;
      001E  001E  00390     .WORD   30, 30           ;
      001E  001E  00394     .WORD   30, 30           ;
      001E  001E  00398     .WORD   30, 30           ;
      001E  001E  0039C     .WORD   30, 30           ;
      001E  001E  003A0     .WORD   30, 30           ;
      001E  001E  003A4     .WORD   30, 30           ;
      001E  001E  003A8     .WORD   30, 30           ;
      001E  001E  003AC     .WORD   30, 30           ;
      001E  001E  003B0     .WORD   30, 30           ;
      001E  001E  003B4      WORD   30, 30           ;
      001E  001E  003B8     .WORD   30, 30           ;
      0021  0021  003BC     .WORD   33, 33           ;
      001E  001E  003C0     .WORD   30, 30           ;
      001E  001E  003C4     .WORD   30, 30           ;
      001E  001E  003C8     .WORD   30, 30           ;
      001E  001E  003CC     .WORD   30, 30           ;
      001E  001E  003D0     .WORD   30, 30           ;
      001E  001E  003D4     .WORD   30, 30           ;
      0003  001E  003D8     .WORD   30, 3            ;
      001F  0000  003DC     .WORD   0, 31            ;
      0020  0000  003E0     .WORD   0, 32            ;
      0021  0000  003E4     .WORD   0, 33            ;
      0022  0000  003E8     .WORD   0, 34            ;
```

```
001E  001E  003EC     .WORD   30, 30          ;
001E  001E  003F0     .WORD   30, 30          ;
001E  001E  003F4     .WORD   30, 30          ;
001E  001E  003F8     .WORD   30, 30          ;
001E  001E  003FC     .WORD   30, 30          ;
001E  001E  00400     .WORD   30, 30          ;
001E  001E  00404     .WORD   30, 30          ;
001E  001E  00408     .WORD   30, 30          ;
001E  001E  0040C     .WORD   30, 30          ;
001E  001E  00410     .WORD   30, 30          ;
001D  001D  00414     .WORD   29, 29          ;
001E  001E  00418     .WORD   30, 30          ;
001E  001E  0041C     .WORD   30, 30          ;
001E  001E  00420     .WORD   30, 30          ;
001E  001E  00424     .WORD   30, 30          ;
001E  001E  00428     .WORD   30, 30          ;
001E  001E  0042C     .WORD   30, 30          ;
001E  001E  00430     .WORD   30, 30          ;
001E  001E  00434     .WORD   30, 30          ;
001E  001E  00438     .WORD   30, 30          ;
001E  001E  0043C     .WORD   30, 30          ;
001E  001E  00440     .WORD   30, 30          ;
001E  001E  00444     .WORD   30, 30          ;
001E  001E  00448     .WORD   30, 30          ;
0021  0021  0044C     .WORD   33, 33          ;
001E  001E  00450     .WORD   30, 30          ;
001E  001E  00454     .WORD   30, 30          ;
001E  001E  00458     .WORD   30, 30          ;
001E  001E  0045C     .WORD   30, 30          ;
001E  001E  00460     .WORD   30, 30          ;
001E  001E  00464     .WORD   30, 30          ;
0003  001E  00468     .WORD   30, 3           ;
001F  0000  0046C     .WORD   0, 31           ;
0020  0000  00470     .WORD   0, 32           ;
0021  0000  00474     .WORD   0, 33           ;
0022  0000  00478     .WORD   0, 34           ;
001E  001E  0047C     .WORD   30, 30          ;
001E  001E  00480     .WORD   30, 30          ;
001E  001E  00484     .WORD   30, 30          ;
001E  001E  00488     .WORD   30, 30          ;
001E  001E  0048C     .WORD   30, 30          ;
001E  001E  00490     .WORD   30, 30          ;
001E  001E  00494     .WORD   30, 30          ;
001E  001E  00498     .WORD   30, 30          ;
001E  001E  0049C     .WORD   30, 30          ;
001E  001E  004A0     .WORD   30, 30          ;
001D  001D  004A4     .WORD   29, 29          ;
001E  001E  004A8     .WORD   30, 30          ;
001E  001E  004AC     .WORD   30, 30          ;
001E  001E  004B0     .WORD   30, 30          ;
001E  001E  004B4     .WORD   30, 30          ;
001E  001E  004B8     .WORD   30, 30          ;
001E  001E  004BC     .WORD   30, 30          ;
001E  001E  004C0     .WORD   30, 30          ;
001E  001E  004C4     .WORD   30, 30          ;
```

```
001E  001E  004C8     .WORD   30, 30          ;
001E  001E  004CC     .WORD   30, 30          ;
001E  001E  004D0     .WORD   30, 30          ;
001E  001E  004D4     .WORD   30, 30          ;
001E  001E  004D8     .WORD   30, 30          ;
0021  0021  004DC     .WORD   33, 33          ;
001E  001E  004E0     .WORD   30, 30          ;
001E  001E  004E4     .WORD   30, 30          ;
001E  001E  004E8     .WORD   30, 30          ;
001E  001E  004EC     .WORD   30, 30          ;
001E  001E  004F0     .WORD   30, 30          ;
001E  001E  004F4     .WORD   30, 30          ;
0003  001E  004F8     .WORD   30, 3           ;
001F  0000  004FC     .WORD   0, 31           ;
0020  0000  00500     .WORD   0, 32           ;
0021  0000  00504     .WORD   0, 33           ;
0022  0000  00508     .WORD   0, 34           ;
001E  001E  0050C     .WORD   30, 30          ;
001E  001E  00510     .WORD   30, 30          ;
001E  001E  00514     .WORD   30, 30          ;
001E  001E  00518     .WORD   30, 30          ;
001E  001E  0051C     .WORD   30, 30          ;
001E  001E  00520     .WORD   30, 30          ;
001E  001E  00524     .WORD   30, 30          ;
001E  001E  00528     .WORD   30, 30          ;
001E  001E  0052C     .WORD   30, 30          ;
001E  001E  00530     .WORD   30, 30          ;
001D  001D  00534     .WORD   29, 29          ;
001E  001E  00538     .WORD   30, 30          ;
001E  001E  0053C     .WORD   30, 30          ;
001E  001E  00540     .WORD   30, 30          ;
001E  001E  00544     .WORD   30, 30          ;
001E  001E  00548     .WORD   30, 30          ;
001E  001E  0054C     .WORD   30, 30          ;
001E  001E  00550     .WORD   30, 30          ;
001E  001E  00554     .WORD   30, 30          ;
001E  001E  00558     .WORD   30, 30          ;
001E  001E  0055C     .WORD   30, 30          ;
001E  001E  00560     .WORD   30, 30          ;
001E  001E  00564     .WORD   30, 30          ;
001E  001E  00568     .WORD   30, 30          ;
001E  001E  0056C     .WORD   30, 30          ;
001E  001E  00570     .WORD   30, 30          ;
001E  001E  00574     .WORD   30, 30          ;
001E  001E  00578     .WORD   30, 30          ;
001E  001E  0057C     .WORD   30, 30          ;
001E  001E  00580     .WORD   30, 30          ;
001E  001E  00584     .WORD   30, 30          ;
0003  001E  00588     .WORD   30, 3           ;
001F  0000  0058C     .WORD   0, 31           ;
0020  0000  00590     .WORD   0, 32           ;
0021  0000  00594     .WORD   0, 33           ;
0022  0000  00598     .WORD   0, 34           ;
001E  001E  0059C     .WORD   30, 30          ;
001E  001E  005A0     .WORD   30, 30          ;
```

```
001E  001E  005A4    .WORD   30, 30          ;
001E  001E  005A8    .WORD   30, 30          ;
001E  001E  005AC    .WORD   30, 30          ;
001E  001E  005B0    .WORD   30, 30          ;
001E  001E  005B4    .WORD   30, 30          ;
001E  001E  005B8    .WORD   30, 30          ;
001E  001E  005BC    .WORD   30, 30          ;
001E  001E  005C0    .WORD   30, 30          ;
001D  001D  005C4    .WORD   29, 29          ;
001E  001E  005C8    .WORD   30, 30          ;
001E  001E  005CC    .WORD   30, 30          ;
001E  001E  005D0    .WORD   30, 30          ;
001E  001E  005D4    .WORD   30, 30          ;
001E  001E  005D8    .WORD   30, 30          ;
001E  001E  005DC    .WORD   30, 30          ;
001E  001E  005E0    .WORD   30, 30          ;
001E  001E  005E4    .WORD   30, 30          ;
001E  001E  005E8    .WORD   30, 30          ;
001E  001E  005EC    .WORD   30, 30          ;
001E  001E  005F0    .WORD   30, 30          ;
001E  001E  005F4    .WORD   30, 30          ;
001E  001E  005F8    .WORD   30, 30          ;
001E  001E  005FC    .WORD   30, 30          ;
001E  001E  00600    .WORD   30, 30          ;
001E  001E  00604    .WORD   30, 30          ;
001E  001E  00608    .WORD   30, 30          ;
001E  001E  0060C    .WORD   30, 30          ;
001E  001E  00610    .WORD   30, 30          ;
001E  001E  00614    .WORD   30, 30          ;
0003  001E  00618    .WORD   30, 3           ;
001F  0000  0061C    .WORD   0, 31           ;
0020  0000  00620    .WORD   0, 32           ;
0021  0000  00624    .WORD   0, 33           ;
0022  0000  00628    .WORD   0, 34           ;
001E  001E  0062C    .WORD   30, 30          ;
001E  001E  00630    .WORD   30, 30          ;
001E  001E  00634    .WORD   30, 30          ;
001E  001E  00638    .WORD   30, 30          ;
001E  001E  0063C    .WORD   30, 30          ;
001E  001E  00640    .WORD   30, 30          ;
001E  001E  00644    .WORD   30, 30          ;
001E  001E  00648    .WORD   30, 30          ;
001D  001D  0064C    .WORD   29, 29          ;
001D  001D  00650    .WORD   29, 29          ;
001D  001D  00654    .WORD   29, 29          ;
001E  001E  00658    .WORD   30, 30          ;
001E  001E  0065C    .WORD   30, 30          ;
001E  001E  00660    .WORD   30, 30          ;
001E  001E  00664    .WORD   30, 30          ;
001E  001E  00668    .WORD   30, 30          ;
001E  001E  0066C    .WORD   30, 30          ;
001E  001E  00670    .WORD   30, 30          ;
001E  001E  00674    .WORD   30, 30          ;
001E  001E  00678    .WORD   30, 30          ;
001F  0001  0067C    .WORD   1, 31           ;
```

```
      0020  0001  00680     .WORD   1, 32           ;
      0020  0001  00684     .WORD   1, 32           ;
      0020  0001  00688     .WORD   1, 32           ;
      0021  0021  0068C     .WORD   33, 33          ;
      001E  001E  00690     .WORD   30, 30          ;
      001E  001E  00694     .WORD   30, 30          ;
      001E  001E  00698     .WORD   30, 30          ;
      001E  001E  0069C     .WORD   30, 30          ;
      001E  001E  006A0     .WORD   30, 30          ;
      001E  001E  006A4     .WORD   30, 30          ;
      0003  001E  006A8     .WORD   30, 3           ;
      001E  001E  006AC     .WORD   30, 30          ;
      001E  001E  006B0     .WORD   30, 30          ;
      001E  001E  006B4     .WORD   30, 30          ;
      001E  001E  006B8     .WORD   30, 30          ;
      001E  001E  006BC     .WORD   30, 30          ;
      001E  001E  006C0     .WORD   30, 30          ;
      001E  001E  006C4     .WORD   30, 30          ;
      001E  001E  006C8     .WORD   30, 30          ;
      001E  001E  006CC     .WORD   30, 30          ;
      001E  001E  006D0     .WORD   30, 30          ;
      001E  001E  006D4     .WORD   30, 30          ;
      001E  001E  006D8     .WORD   30, 30          ;
      001D  001D  006DC     .WORD   29, 29          ;
      001D  001D  006E0     .WORD   29, 29          ;
      001D  001D  006E4     .WORD   29, 29          ;
      001E  001E  006E8     .WORD   30, 30          ;
      001E  001E  006EC     .WORD   30, 30          ;
      001E  001E  006F0     .WORD   30, 30          ;
      001E  001E  006F4     .WORD   30, 30          ;
      001E  001E  006F8     .WORD   30, 30          ;
      001E  001E  006FC     .WORD   30, 30          ;
      001E  001E  00700     .WORD   30, 30          ;
      001E  001E  00704     .WORD   30, 30          ;
      001E  001E  00708     .WORD   30, 30          ;
      001F  0001  0070C     .WORD   1, 31           ;
      0020  0001  00710     .WORD   1, 32           ;
      0020  0001  00714     .WORD   1, 32           ;
      0020  0001  00718     .WORD   1, 32           ;
      0021  0021  0071C     .WORD   33, 33          ;
      001E  001E  00720     .WORD   30, 30          ;
      001E  001E  00724     .WORD   30, 30          ;
      001E  001E  00728     .WORD   30, 30          ;
      001E  001E  0072C     .WORD   30, 30          ;
      001E  001E  00730     .WORD   30, 30          ;
      001E  001E  00734     .WORD   30, 30          ;
      0003  001E  00738     .WORD   30, 3           ;
      001F  0000  0073C     .WORD   0, 31           ;
      0020  0000  00740     .WORD   0, 32           ;
      0021  0000  00744     .WORD   0, 33           ;
      0022  0000  00748     .WORD   0, 34           ;
      001E  001E  0074C     .WORD   30, 30          ;
      001E  001E  00750     .WORD   30, 30          ;
      001E  001E  00754     .WORD   30, 30          ;
      001E  001E  00758     .WORD   30, 30          ;
```

```
      001E  001E  0075C     .WORD   30, 30          ;
      001E  001E  00760     .WORD   30, 30          ;
      001E  001E  00764     .WORD   30, 30          ;
      001E  001E  00768     .WORD   30, 30          ;
      001E  001E  0076C     .WORD   30, 30          ;
      001E  001E  00770     .WORD   30, 30          ;
      001D  001D  00774     .WORD   29, 29          ;
      001E  001E  00778     .WORD   30, 30          ;
      001E  001E  0077C     .WORD   30, 30          ;
      001E  001E  00780     .WORD   30, 30          ;
      001E  001E  00784     .WORD   30, 30          ;
      001E  001E  00788     .WORD   30, 30          ;
      001E  001E  0078C     .WORD   30, 30          ;
      001E  001E  00790     .WORD   30, 30          ;
      001E  001E  00794     .WORD   30, 30          ;
      001E  001E  00798     .WORD   30, 30          ;
      001E  001E  0079C     .WORD   30, 30          ;
      001E  001E  007A0     .WORD   30, 30          ;
      001E  001E  007A4     .WORD   30, 30          ;
      001E  001E  007A8     .WORD   30, 30          ;
      0021  0021  007AC     .WORD   33, 33          ;
      001E  001E  007B0     .WORD   30, 30          ;
      001E  001E  007B4     .WORD   30, 30          ;
      001E  001E  007B8     .WORD   30, 30          ;
      001E  001E  007BC     .WORD   30, 30          ;
      001E  001E  007C0     .WORD   30, 30          ;
      001E  001E  007C4     .WORD   30, 30          ;
      0003  001E  007C8     .WORD   30, 3           ;
      001F  0000  007CC     .WORD   0, 31           ;
      0020  0000  007D0     .WORD   0, 32           ;
      0021  0000  007D4     .WORD   0, 33           ;
      0022  0000  007D8     .WORD   0, 34           ;
      001E  001E  007DC     .WORD   30, 30          ;
      001E  001E  007E0     .WORD   30, 30          ;
      001E  001E  007E4     .WORD   30, 30          ;
      001E  001E  007E8     .WORD   30, 30          ;
      001E  001E  007EC     .WORD   30, 30          ;
      001E  001E  007F0     .WORD   30, 30          ;
      001E  001E  007F4     .WORD   30, 30          ;
      001E  001E  007F8     .WORD   30, 30          ;
      001E  001E  007FC     .WORD   30, 30          ;
      001E  001E  00800     .WORD   30, 30          ;
      0009  0000  00804     .WORD   0, 9          ;
      001E  001E  00808     .WORD   30, 30          ;
      001E  001E  0080C     .WORD   30, 30          ;
      001E  001E  00810     .WORD   30, 30          ;
      001E  001E  00814     .WORD   30, 30          ;
      001E  001E  00818     .WORD   30, 30          ;
      001E  001E  0081C     .WORD   30, 30          ;
      001E  001E  00820     .WORD   30, 30          ;
      001E  001E  00824     .WORD   30, 30          ;
      001E  001E  00828     .WORD   30, 30          ;
      001E  001E  0082C     .WORD   30, 30          ;
      001E  001E  00830     .WORD   30, 30          ;
      001E  001E  00834     .WORD   30, 30          ;
```

```
001E  001E  00838     .WORD   30, 30          ;
0017  0000  0083C     .WORD   0, 23           ;
001E  001E  00840     .WORD   30, 30          ;
001E  001E  00844     .WORD   30, 30          ;
001E  001E  00848     .WORD   30, 30          ;
001E  001E  0084C     .WORD   30, 30          ;
001E  001E  00850     .WORD   30, 30          ;
001E  001E  00854     .WORD   30, 30          ;
0003  001E  00858     .WORD   30, 3           ;
001F  0000  0085C     .WORD   0, 31           ;
0020  0000  00860     .WORD   0, 32           ;
0021  0000  00864     .WORD   0, 33           ;
0022  0000  00868     .WORD   0, 34           ;
001E  001E  0086C     .WORD   30, 30          ;
001E  001E  00870     .WORD   30, 30          ;
001E  001E  00874     .WORD   30, 30          ;
001E  001E  00878     .WORD   30, 30          ;
001E  001E  0087C     .WORD   30, 30          ;
001E  001E  00880     .WORD   30, 30          ;
001E  001E  00884     .WORD   30, 30          ;
001E  001E  00888     .WORD   30, 30          ;
001E  001E  0088C     .WORD   30, 30          ;
001E  001E  00890     .WORD   30, 30          ;
0009  0000  00894     .WORD   0, 9            ;
001E  001E  00898     .WORD   30, 30          ;
001E  001E  0089C     .WORD   30, 30          ;
001E  001E  008A0     .WORD   30, 30          ;
001E  001E  008A4     .WORD   30, 30          ;
001E  001E  008A8     .WORD   30, 30          ;
001E  001E  008AC     .WORD   30, 30          ;
001E  001E  008B0     .WORD   30, 30          ;
001E  001E  008B4     .WORD   30, 30          ;
001E  001E  008B8     .WORD   30, 30          ;
001E  001E  008BC     .WORD   30, 30          ;
001E  001E  008C0     .WORD   30, 30          ;
001E  001E  008C4     .WORD   30, 30          ;
001E  001E  008C8     .WORD   30, 30          ;
0017  0000  008CC     .WORD   0, 23           ;
001E  001E  008D0     .WORD   30, 30          ;
001E  001E  008D4     .WORD   30, 30          ;
001E  001E  008D8     .WORD   30, 30          ;
001E  001E  008DC     .WORD   30, 30          ;
001E  001E  008E0     .WORD   30, 30          ;
001E  001E  008E4     .WORD   30, 30          ;
0003  001E  008E8     .WORD   30, 3           ;
001E  001E  008EC     .WORD   30, 30          ;
001E  001E  008F0     .WORD   30, 30          ;
001E  001E  008F4     .WORD   30, 30          ;
001E  001E  008F8     .WORD   30, 30          ;
001E  001E  008FC     .WORD   30, 30          ;
001E  001E  00900     .WORD   30, 30          ;
001E  001E  00904     .WORD   30, 30          ;
001E  001E  00908     .WORD   30, 30          ;
001E  001E  0090C     .WORD   30, 30          ;
001E  001E  00910     .WORD   30, 30          ;
```

```
001E  001E  00914     .WORD   30, 30          ;
001E  001E  00918     .WORD   30, 30          ;
001E  001E  0091C     .WORD   30, 30          ;
001E  001E  00920     .WORD   30, 30          ;
0009  0000  00924     .WORD   0, 9           ;
001E  001E  00928     .WORD   30, 30          ;
001E  001E  0092C     .WORD   30, 30          ;
001E  001E  00930     .WORD   30, 30          ;
001E  001E  00934     .WORD   30, 30          ;
001E  001E  00938     .WORD   30, 30          ;
001E  001E  0093C     .WORD   30, 30          ;
001E  001E  00940     .WORD   30, 30          ;
001E  001E  00944     .WORD   30, 30          ;
001E  001E  00948     .WORD   30, 30          ;
001E  001E  0094C     .WORD   30, 30          ;
001E  001E  00950     .WORD   30, 30          ;
001E  001E  00954     .WORD   30, 30          ;
001E  001E  00958     .WORD   30, 30          ;
0017  0000  0095C     .WORD   0, 23           ;
001E  001E  00960     .WORD   30, 30          ;
001E  001E  00964     .WORD   30, 30          ;
001E  001E  00968     .WORD   30, 30          ;
001E  001E  0096C     .WORD   30, 30          ;
001E  001E  00970     .WORD   30, 30          ;
001E  001E  00974     .WORD   30, 30          ;
0003  001E  00978     .WORD   30, 3           ;
001F  0000  0097C     .WORD   0, 31           ;
0020  0000  00980     .WORD   0, 32           ;
0021  0000  00984     .WORD   0, 33           ;
0022  0000  00988     .WORD   0, 34           ;
001E  001E  0098C     .WORD   30, 30          ;
001E  001E  00990     .WORD   30, 30          ;
001E  001E  00994     .WORD   30, 30          ;
001E  001E  00998     .WORD   30, 30          ;
001E  001E  0099C     .WORD   30, 30          ;
001E  001E  009A0     .WORD   30, 30          ;
001E  001E  009A4     .WORD   30, 30          ;
001E  001E  009A8     .WORD   30, 30          ;
001E  001E  009AC     .WORD   30, 30          ;
001E  001E  009B0     .WORD   30, 30          ;
0009  0000  009B4     .WORD   0, 9           ;
001E  001E  009B8     .WORD   30, 30          ;
001E  001E  009BC     .WORD   30, 30          ;
001E  001E  009C0     .WORD   30, 30          ;
001E  001E  009C4     .WORD   30, 30          ;
001E  001E  009C8     .WORD   30, 30          ;
001E  001E  009CC     .WORD   30, 30          ;
001E  001E  009D0     .WORD   30, 30          ;
001E  001E  009D4     .WORD   30, 30          ;
001E  001E  009D8     .WORD   30, 30          ;
001E  001E  009DC     .WORD   30, 30          ;
001E  001E  009E0     .WORD   30, 30          ;
001E  001E  009E4     .WORD   30, 30          ;
001E  001E  009E8     .WORD   30, 30          ;
0017  0000  009EC     .WORD   0, 23           ;
```

```
001E  001E  009F0     .WORD   30, 30          ;
001E  001E  009F4     .WORD   30, 30          ;
001E  001E  009F8     .WORD   30, 30          ;
001E  001E  009FC     .WORD   30, 30          ;
001E  001E  00A00     .WORD   30, 30          ;
001E  001E  00A04     .WORD   30, 30          ;
0003  001E  00A08     .WORD   30, 3           ;
001E  001E  00A0C     .WORD   30, 30          ;
001E  001E  00A10     .WORD   30, 30          ;
001E  001E  00A14     .WORD   30, 30          ;
001E  001E  00A18     .WORD   30, 30          ;
001E  001E  00A1C     .WORD   30, 30          ;
001E  001E  00A20     .WORD   30, 30          ;
001E  001E  00A24     .WORD   30, 30          ;
001E  001E  00A28     .WORD   30, 30          ;
001E  001E  00A2C     .WORD   30, 30          ;
001E  001E  00A30     .WORD   30, 30          ;
001E  001E  00A34     .WORD   30, 30          ;
001E  001E  00A38     .WORD   30, 30          ;
001E  001E  00A3C     .WORD   30, 30          ;
001E  001E  00A40     .WORD   30, 30          ;
001D  001D  00A44     .WORD   29, 29          ;
001E  001E  00A48     .WORD   30, 30          ;
001E  001E  00A4C     .WORD   30, 30          ;
001E  001E  00A50     .WORD   30, 30          ;
001E  001E  00A54     .WORD   30, 30          ;
001E  001E  00A58     .WORD   30, 30          ;
001E  001E  00A5C     .WORD   30, 30          ;
001E  001E  00A60     .WORD   30, 30          ;
001E  001E  00A64     .WORD   30, 30          ;
001E  001E  00A68     .WORD   30, 30          ;
001E  001E  00A6C     .WORD   30, 30          ;
001E  001E  00A70     .WORD   30, 30          ;
001E  001E  00A74     .WORD   30, 30          ;
001E  001E  00A78     .WORD   30, 30          ;
0021  0021  00A7C     .WORD   33, 33          ;
001E  001E  00A80     .WORD   30, 30          ;
001E  001E  00A84     .WORD   30, 30          ;
001E  001E  00A88     .WORD   30, 30          ;
001E  001E  00A8C     .WORD   30, 30          ;
001E  001E  00A90     .WORD   30, 30          ;
001E  001E  00A94     .WORD   30, 30          ;
0003  001E  00A98     .WORD   30, 3           ;
001F  0000  00A9C     .WORD   0, 31           ;
0020  0000  00AA0     .WORD   0, 32           ;
0021  0000  00AA4     .WORD   0, 33           ;
0022  0000  00AA8     .WORD   0, 34           ;
001E  001E  00AAC     .WORD   30, 30          ;
001E  001E  00AB0     .WORD   30, 30          ;
001E  001E  00AB4     .WORD   30, 30          ;
001E  001E  00AB8     .WORD   30, 30          ;
001E  001E  00ABC     .WORD   30, 30          ;
001E  001E  00AC0     .WORD   30, 30          ;
001E  001E  00AC4     .WORD   30, 30          ;
001E  001E  00AC8     .WORD   30, 30          ;
```

```
001E  001E  00ACC     .WORD   30, 30          ;
001E  001E  00AD0     .WORD   30, 30          ;
001D  001D  00AD4     .WORD   29, 29          ;
001E  001E  00AD8     .WORD   30, 30          ;
001E  001E  00ADC     .WORD   30, 30          ;
001E  001E  00AE0     .WORD   30, 30          ;
001E  001E  00AE4     .WORD   30, 30          ;
001E  001E  00AE8     .WORD   30, 30          ;
001E  001E  00AEC     .WORD   30, 30          ;
001E  001E  00AF0     .WORD   30, 30          ;
001E  001E  00AF4     .WORD   30, 30          ;
001E  001E  00AF8     .WORD   30, 30          ;
001E  001E  00AFC     .WORD   30, 30          ;
001E  001E  00B00     .WORD   30, 30          ;
001E  001E  00B04     .WORD   30, 30          ;
001E  001E  00B08     .WORD   30, 30          ;
0021  0021  00B0C     .WORD   33, 33          ;
001E  001E  00B10     .WORD   30, 30          ;
001E  001E  00B14     .WORD   30, 30          ;
001E  001E  00B18     .WORD   30, 30          ;
001E  001E  00B1C     .WORD   30, 30          ;
001E  001E  00B20     .WORD   30, 30          ;
001E  001E  00B24     .WORD   30, 30          ;
0003  001E  00B28     .WORD   30, 3           ;
001F  0000  00B2C     .WORD   0, 31           ;
0020  0000  00B30     .WORD   0, 32           ;
0021  0000  00B34     .WORD   0, 33           ;
0022  0000  00B38     .WORD   0, 34           ;
001E  001E  00B3C     .WORD   30, 30          ;
001E  001E  00B40     .WORD   30, 30          ;
001E  001E  00B44     .WORD   30, 30          ;
001E  001E  00B48     .WORD   30, 30          ;
001E  001E  00B4C     .WORD   30, 30          ;
001E  001E  00B50     .WORD   30, 30          ;
001E  001E  00B54     .WORD   30, 30          ;
001E  001E  00B58     .WORD   30, 30          ;
001E  001E  00B5C     .WORD   30, 30          ;
001E  001E  00B60     .WORD   30, 30          ;
001D  001D  00B64     .WORD   29, 29          ;
001E  001E  00B68     .WORD   30, 30          ;
001E  001E  00B6C     .WORD   30, 30          ;
001E  001E  00B70     .WORD   30, 30          ;
001E  001E  00B74     .WORD   30, 30          ;
001E  001E  00B78     .WORD   30, 30          ;
001E  001E  00B7C     .WORD   30, 30          ;
001E  001E  00B80     .WORD   30, 30          ;
001E  001E  00B84     .WORD   30, 30          ;
001E  001E  00B88     .WORD   30, 30          ;
001E  001E  00B8C     .WORD   30, 30          ;
001E  001E  00B90     .WORD   30, 30          ;
001E  001E  00B94     .WORD   30, 30          ;
001E  001E  00B98     .WORD   30, 30          ;
0021  0021  00B9C     .WORD   33, 33          ;
001E  001E  00BA0     .WORD   30, 30          ;
001E  001E  00BA4     .WORD   30, 30          ;
```

```
      001E  001E  00BA8     .WORD   30, 30          ;
      001E  001E  00BAC     .WORD   30, 30          ;
      001E  001E  00BB0     .WORD   30, 30          ;
      001D  0000  00BB4     .WORD   0, 29           ;
      0003  001E  00BB8     .WORD   30, 3           ;
      001F  0000  00BBC     .WORD   0, 31           ;
      0020  0000  00BC0     .WORD   0, 32           ;
      0021  0000  00BC4     .WORD   0, 33           ;
      0022  0000  00BC8     .WORD   0, 34           ;
      001E  001E  00BCC     .WORD   30, 30          ;
      001E  001E  00BD0     .WORD   30, 30          ;
      001E  001E  00BD4     .WORD   30, 30          ;
      001E  001E  00BD8     .WORD   30, 30          ;
      001E  001E  00BDC     .WORD   30, 30          ;
      001E  001E  00BE0     .WORD   30, 30          ;
      001E  001E  00BE4     .WORD   30, 30          ;
      001E  001E  00BE8     .WORD   30, 30          ;
      001D  001D  00BEC     .WORD   29, 29          ;
      001D  001D  00BF0     .WORD   29, 29          ;
      001D  001D  00BF4     .WORD   29, 29          ;
      001E  001E  00BF8     .WORD   30, 30          ;
      001E  001E  00BFC     .WORD   30, 30          ;
      001E  001E  00C00     .WORD   30, 30          ;
      001E  001E  00C04     .WORD   30, 30          ;
      001E  001E  00C08     .WORD   30, 30          ;
      001E  001E  00C0C     .WORD   30, 30          ;
      001E  001E  00C10     .WORD   30, 30          ;
      001E  001E  00C14     .WORD   30, 30          ;
      001E  001E  00C18     .WORD   30, 30          ;
      001F  0001  00C1C     .WORD   1, 31           ;
      0020  0001  00C20     .WORD   1, 32           ;
      0020  0001  00C24     .WORD   1, 32           ;
      0020  0001  00C28     .WORD   1, 32           ;
      0020  0001  00C2C     .WORD   1, 32           ;
      001E  001E  00C30     .WORD   30, 30          ;
      001E  001E  00C34     .WORD   30, 30          ;
      001E  001E  00C38     .WORD   30, 30          ;
      001E  001E  00C3C     .WORD   30, 30          ;
      001E  001E  00C40     .WORD   30, 30          ;
      001E  001E  00C44     .WORD   30, 30          ;
      0003  001E  00C48     .WORD   30, 3           ;
      001F  0000  00C4C     .WORD   0, 31           ;
      0020  0000  00C50     .WORD   0, 32           ;
      0021  0000  00C54     .WORD   0, 33           ;
      0022  0000  00C58     .WORD   0, 34           ;
      001E  001E  00C5C     .WORD   30, 30          ;
      001E  001E  00C60     .WORD   30, 30          ;
      001E  001E  00C64     .WORD   30, 30          ;
      001E  001E  00C68     .WORD   30, 30          ;
      001E  001E  00C6C     .WORD   30, 30          ;
      001E  001E  00C70     .WORD   30, 30          ;
      001E  001E  00C74     .WORD   30, 30          ;
      001E  001E  00C78     .WORD   30, 30          ;
      001D  001D  00C7C     .WORD   29, 29          ;
      001D  001D  00C80     .WORD   29, 29          ;
```

```
001D  001D  00C84     .WORD   29, 29          ;
001E  001E  00C88     .WORD   30, 30          ;
001E  001E  00C8C     .WORD   30, 30          ;
001E  001E  00C90     .WORD   30, 30          ;
001E  001E  00C94     .WORD   30, 30          ;
001E  001E  00C98     .WORD   30, 30          ;
001E  001E  00C9C     .WORD   30, 30          ;
001E  001E  00CA0     .WORD   30, 30          ;
001E  001E  00CA4     .WORD   30, 30          ;
001E  001E  00CA8     .WORD   30, 30          ;
001F  0001  00CAC     .WORD   1, 31           ;
0020  0001  00CB0     .WORD   1, 32           ;
0020  0001  00CB4     .WORD   1, 32           ;
0020  0001  00CB8     .WORD   1, 32           ;
0020  0001  00CBC     .WORD   1, 32           ;
001E  001E  00CC0     .WORD   30, 30          ;
001E  001E  00CC4     .WORD   30, 30          ;
001E  001E  00CC8     .WORD   30, 30          ;
001E  001E  00CCC     .WORD   30, 30          ;
001E  001E  00CD0     .WORD   30, 30          ;
001E  001E  00CD4     .WORD   30, 30          ;
0003  001E  00CD8     .WORD   30, 3           ;
001F  0000  00CDC     .WORD   0, 31           ;
0020  0000  00CE0     .WORD   0, 32           ;
0021  0000  00CE4     .WORD   0, 33           ;
0022  0000  00CE8     .WORD   0, 34           ;
001E  001E  00CEC     .WORD   30, 30          ;
001E  001E  00CF0     .WORD   30, 30          ;
001E  001E  00CF4     .WORD   30, 30          ;
001E  001E  00CF8     .WORD   30, 30          ;
001E  001E  00CFC     .WORD   30, 30          ;
001E  001E  00D00     .WORD   30, 30          ;
001E  001E  00D04     .WORD   30, 30          ;
001E  001E  00D08     .WORD   30, 30          ;
001E  001E  00D0C     .WORD   30, 30          ;
001E  001E  00D10     .WORD   30, 30          ;
0009  0000  00D14     .WORD   0, 9            ;
001E  001E  00D18     .WORD   30, 30          ;
001E  001E  00D1C     .WORD   30, 30          ;
001E  001E  00D20     .WORD   30, 30          ;
001E  001E  00D24     .WORD   30, 30          ;
001E  001E  00D28     .WORD   30, 30          ;
001E  001E  00D2C     .WORD   30, 30          ;
001E  001E  00D30     .WORD   30, 30          ;
001E  001E  00D34     .WORD   30, 30          ;
001E  001E  00D38     .WORD   30, 30          ;
001E  001E  00D3C     .WORD   30, 30          ;
001E  001E  00D40     .WORD   30, 30          ;
001E  001E  00D44     .WORD   30, 30          ;
001E  001E  00D48     .WORD   30, 30          ;
0017  0000  00D4C     .WORD   0, 23           ;
001E  001E  00D50     .WORD   30, 30          ;
001E  001E  00D54     .WORD   30, 30          ;
001E  001E  00D58     .WORD   30, 30          ;
001E  001E  00D5C     .WORD   30, 30          ;
```

```
001E  001E  00D60   .WORD   30, 30          ;
001E  001E  00D64   .WORD   30, 30          ;
0003  001E  00D68   .WORD   30, 3           ;
001E  001E  00D6C   .WORD   30, 30          ;
001E  001E  00D70   .WORD   30, 30          ;
001E  001E  00D74   .WORD   30, 30          ;
001E  001E  00D78   .WORD   30, 30          ;
001E  001E  00D7C   .WORD   30, 30          ;
001E  001E  00D80   .WORD   30, 30          ;
001E  001E  00D84   .WORD   30, 30          ;
001E  001E  00D88   .WORD   30, 30          ;
001E  001E  00D8C   .WORD   30, 30          ;
001E  001E  00D90   .WORD   30, 30          ;
001E  001E  00D94   .WORD   30, 30          ;
001E  001E  00D98   .WORD   30, 30          ;
001E  001E  00D9C   .WORD   30, 30          ;
001E  001E  00DA0   .WORD   30, 30          ;
001D  001D  00DA4   .WORD   29, 29          ;
001E  001E  00DA8   .WORD   30, 30          ;
001E  001E  00DAC   .WORD   30, 30          ;
001E  001E  00DB0   .WORD   30, 30          ;
001E  001E  00DB4   .WORD   30, 30          ;
001E  001E  00DB8   .WORD   30, 30          ;
001E  001E  00DBC   .WORD   30, 30          ;
001E  001E  00DC0   .WORD   30, 30          ;
001E  001E  00DC4   .WORD   30, 30          ;
001E  001E  00DC8   .WORD   30, 30          ;
001E  001E  00DCC   .WORD   30, 30          ;
001E  001E  00DD0   .WORD   30, 30          ;
001E  001E  00DD4   .WORD   30, 30          ;
001E  001E  00DD8   .WORD   30, 30          ;
0020  0001  00DDC   .WORD   1, 32           ;
001E  001E  00DE0   .WORD   30, 30          ;
001E  001E  00DE4   .WORD   30, 30          ;
001E  001E  00DE8   .WORD   30, 30          ;
001E  001E  00DEC   .WORD   30, 30          ;
001E  001E  00DF0   .WORD   30, 30          ;
001E  001E  00DF4   .WORD   30, 30          ;
0003  001E  00DF8   .WORD   30, 3           ;
001E  001E  00DFC   .WORD   30, 30          ;
001E  001E  00E00   .WORD   30, 30          ;
001E  001E  00E04   .WORD   30, 30          ;
001E  001E  00E08   .WORD   30, 30          ;
001E  001E  00E0C   .WORD   30, 30          ;
001E  001E  00E10   .WORD   30, 30          ;
001E  001E  00E14   .WORD   30, 30          ;
001E  001E  00E18   .WORD   30, 30          ;
001E  001E  00E1C   .WORD   30, 30          ;
001E  001E  00E20   .WORD   30, 30          ;
001E  001E  00E24   .WORD   30, 30          ;
001E  001E  00E28   .WORD   30, 30          ;
001E  001E  00E2C   .WORD   30, 30          ;
001E  001E  00E30   .WORD   30, 30          ;
001D  001D  00E34   .WORD   29, 29          ;
001E  001E  00E38   .WORD   30, 30          ;
```

```
001E  001E  00E3C    .WORD   30, 30          ;
001E  001E  00E40    .WORD   30, 30          ;
001E  001E  00E44    .WORD   30, 30          ;
001E  001E  00E48    .WORD   30, 30          ;
001E  001E  00E4C    .WORD   30, 30          ;
001E  001E  00E50    .WORD   30, 30          ;
001E  001E  00E54    .WORD   30, 30          ;
001E  001E  00E58    .WORD   30, 30          ;
001E  001E  00E5C    .WORD   30, 30          ;
001E  001E  00E60    .WORD   30, 30          ;
001E  001E  00E64    .WORD   30, 30          ;
001E  001E  00E68    .WORD   30, 30          ;
0021  0021  00E6C    .WORD   33, 33          ;
001E  001E  00E70    .WORD   30, 30          ;
001E  001E  00E74    .WORD   30, 30          ;
001E  001E  00E78    .WORD   30, 30          ;
001E  001E  00E7C    .WORD   30, 30          ;
001E  001E  00E80    .WORD   30, 30          ;
001E  001E  00E84    .WORD   30, 30          ;
0003  001E  00E88    .WORD   30, 3           ;
001F  0000  00E8C    .WORD   0, 31           ;
0020  0000  00E90    .WORD   0, 32           ;
0021  0000  00E94    .WORD   0, 33           ;
0022  0000  00E98    .WORD   0, 34           ;
001E  001E  00E9C    .WORD   30, 30          ;
001E  001E  00EA0    .WORD   30, 30          ;
001E  001E  00EA4    .WORD   30, 30          ;
001E  001E  00EA8    .WORD   30, 30          ;
001E  001E  00EAC    .WORD   30, 30          ;
001E  001E  00EB0    .WORD   30, 30          ;
001E  001E  00EB4    .WORD   30, 30          ;
001E  001E  00EB8    .WORD   30, 30          ;
001E  001E  00EBC    .WORD   30, 30          ;
001E  001E  00EC0    .WORD   30, 30          ;
001E  001E  00EC4    .WORD   30, 30          ;
001E  001E  00EC8    .WORD   30, 30          ;
001E  001E  00ECC    .WORD   30, 30          ;
001E  001E  00ED0    .WORD   30, 30          ;
001E  001E  00ED4    .WORD   30, 30          ;
001E  001E  00ED8    .WORD   30, 30          ;
001E  001E  00EDC    .WORD   30, 30          ;
001E  001E  00EE0    .WORD   30, 30          ;
001E  001E  00EE4    .WORD   30, 30          ;
001E  001E  00EE8    .WORD   30, 30          ;
001E  001E  00EEC    .WORD   30, 30          ;
001E  001E  00EF0    .WORD   30, 30          ;
001E  001E  00EF4    .WORD   30, 30          ;
001E  001E  00EF8    .WORD   30, 30          ;
001E  001E  00EFC    .WORD   30, 30          ;
001E  001E  00F00    .WORD   30, 30          ;
001E  001E  00F04    .WORD   30, 30          ;
001E  001E  00F08    .WORD   30, 30          ;
001E  001E  00F0C    .WORD   30, 30          ;
001E  001E  00F10    .WORD   30, 30          ;
001E  001E  00F14    .WORD   30, 30          ;
```

```
001E  001E  00F18     .WORD   30, 30          ;
001E  001E  00F1C     .WORD   30, 30          ;
001E  001E  00F20     .WORD   30, 30          ;
001E  001E  00F24     .WORD   30, 30          ;
001E  001E  00F28     .WORD   30, 30          ;
001E  001E  00F2C     .WORD   30, 30          ;
001E  001E  00F30     .WORD   30, 30          ;
001E  001E  00F34     .WORD   30, 30          ;
001E  001E  00F38     .WORD   30, 30          ;
001E  001E  00F3C     .WORD   30, 30          ;
001E  001E  00F40     .WORD   30, 30          ;
001E  001E  00F44     .WORD   30, 30          ;
001E  001E  00F48     .WORD   30, 30          ;
001E  001E  00F4C     .WORD   30, 30          ;
001E  001E  00F50     .WORD   30, 30          ;
001D  001D  00F54     .WORD   29, 29          ;
001E  001E  00F58     .WORD   30, 30          ;
001E  001E  00F5C     .WORD   30, 30          ;
001E  001E  00F60     .WORD   30, 30          ;
001E  001E  00F64     .WORD   30, 30          ;
001E  001E  00F68     .WORD   30, 30          ;
001E  001E  00F6C     .WORD   30, 30          ;
001E  001E  00F70     .WORD   30, 30          ;
001E  001E  00F74     .WORD   30, 30          ;
001E  001E  00F78     .WORD   30, 30          ;
001E  001E  00F7C     .WORD   30, 30          ;
001E  001E  00F80     .WORD   30, 30          ;
001E  001E  00F84     .WORD   30, 30          ;
001E  001E  00F88     .WORD   30, 30          ;
0022  0022  00F8C     .WORD   34, 34          ;
001E  001E  00F90     .WORD   30, 30          ;
001E  001E  00F94     .WORD   30, 30          ;
001E  001E  00F98     .WORD   30, 30          ;
001E  001E  00F9C     .WORD   30, 30          ;
001E  001E  00FA0     .WORD   30, 30          ;
001E  001E  00FA4     .WORD   30, 30          ;
0003  001E  00FA8     .WORD   30, 3           ;
001F  0000  00FAC     .WORD   0, 31           ;
0020  0000  00FB0     .WORD   0, 32           ;
0021  0000  00FB4     .WORD   0, 33           ;
0022  0000  00FB8     .WORD   0, 34           ;
001E  001E  00FBC     .WORD   30, 30          ;
001E  001E  00FC0     .WORD   30, 30          ;
001E  001E  00FC4     .WORD   30, 30          ;
001E  001E  00FC8     .WORD   30, 30          ;
001E  001E  00FCC     .WORD   30, 30          ;
001E  001E  00FD0     .WORD   30, 30          ;
001E  001E  00FD4     .WORD   30, 30          ;
001E  001E  00FD8     .WORD   30, 30          ;
001E  001E  00FDC     .WORD   30, 30          ;
001E  001E  00FE0     .WORD   30, 30          ;
001D  001D  00FE4     .WORD   29, 29          ;
001E  001E  00FE8     .WORD   30, 30          ;
001E  001E  00FEC     .WORD   30, 30          ;
001E  001E  00FF0     .WORD   30, 30          ;
```

```
      001E  001E  00FF4    .WORD   30, 30          ;
      001E  001E  00FF8    .WORD   30, 30          ;
      001E  001E  00FFC    .WORD   30, 30          ;
      001E  001E  01000    .WORD   30, 30          ;
      001E  001E  01004    .WORD   30, 30          ;
      001E  001E  01008    .WORD   30, 30          ;
      001E  001E  0100C    .WORD   30, 30          ;
      001E  001E  01010    .WORD   30, 30          ;
      001E  001E  01014    .WORD   30, 30          ;
      001E  001E  01018    .WORD   30, 30          ;
      0021  0021  0101C    .WORD   33, 33          ;
      001E  001E  01020    .WORD   30, 30          ;
      001E  001E  01024    .WORD   30, 30          ;
      001E  001E  01028    .WORD   30, 30          ;
      001E  001E  0102C    .WORD   30, 30          ;
      001E  001E  01030    .WORD   30, 30          ;
      001E  001E  01034    .WORD   30, 30          ;
      0003  001E  01038    .WORD   30, 3           ;
      001F  0000  0103C    .WORD   0, 31           ;
      0020  0000  01040    .WORD   0, 32           ;
      0021  0000  01044    .WORD   0, 33           ;
      0022  0000  01048    .WORD   0, 34           ;
      001E  001E  0104C    .WORD   30, 30          ;
      001E  001E  01050    .WORD   30, 30          ;
      001E  001E  01054    .WORD   30, 30          ;
      001E  001E  01058    .WORD   30, 30          ;
      0003  0000  0105C    .WORD   0, 3            ;
      001E  001E  01060    .WORD   30, 30          ;
      001E  001E  01064    .WORD   30, 30          ;
      001E  001E  01068    .WORD   30, 30          ;
      001E  001E  0106C    .WORD   30, 30          ;
      001E  001E  01070    .WORD   30, 30          ;
      001E  001E  01074    .WORD   30, 30          ;
      001E  001E  01078    .WORD   30, 30          ;
      001E  001E  0107C    .WORD   30, 30          ;
      001E  001E  01080    .WORD   30, 30          ;
      001E  001E  01084    .WORD   30, 30          ;
      001E  001E  01088    .WORD   30, 30          ;
      001E  001E  0108C    .WORD   30, 30          ;
      001E  001E  01090    .WORD   30, 30          ;
      001E  001E  01094    .WORD   30, 30          ;
      001E  001E  01098    .WORD   30, 30          ;
      001E  001E  0109C    .WORD   30, 30          ;
      001E  001E  010A0    .WORD   30, 30          ;
      001E  001E  010A4    .WORD   30, 30          ;
      001E  001E  010A8    .WORD   30, 30          ;
      001E  001E  010AC    .WORD   30, 30          ;
      001E  001E  010B0    .WORD   30, 30          ;
      001E  001E  010B4    .WORD   30, 30          ;
      001E  001E  010B8    .WORD   30, 30          ;
      001E  001E  010BC    .WORD   30, 30          ;
      001E  001E  010C0    .WORD   30, 30          ;
      001E  001E  010C4    .WORD   30, 30          ;
      001E  001E  010C8    .WORD   30, 30          ;
      001E  001E  010CC    .WORD   30, 30          ;
```

```
001E  001E  010D0     .WORD   30, 30          ;
001E  001E  010D4     .WORD   30, 30          ;
001E  001E  010D8     .WORD   30, 30          ;
001E  001E  010DC     .WORD   30, 30          ;
001E  001E  010E0     .WORD   30, 30          ;
001E  001E  010E4     .WORD   30, 30          ;
001E  001E  010E8     .WORD   30, 30          ;
001E  001E  010EC     .WORD   30, 30          ;
001E  001E  010F0     .WORD   30, 30          ;
001E  001E  010F4     .WORD   30, 30          ;
001E  001E  010F8     .WORD   30, 30          ;
001D  001D  010FC     .WORD   29, 29          ;
001D  001D  01100     .WORD   29, 29          ;
001D  001D  01104     .WORD   29, 29          ;
001E  001E  01108     .WORD   30, 30          ;
001E  001E  0110C     .WORD   30, 30          ;
001E  001E  01110     .WORD   30, 30          ;
001E  001E  01114     .WORD   30, 30          ;
001E  001E  01118     .WORD   30, 30          ;
001E  001E  0111C     .WORD   30, 30          ;
001E  001E  01120     .WORD   30, 30          ;
001E  001E  01124     .WORD   30, 30          ;
001E  001E  01128     .WORD   30, 30          ;
001F  0001  0112C     .WORD   1, 31           ;
0020  0001  01130     .WORD   1, 32           ;
0020  0001  01134     .WORD   1, 32           ;
0020  0001  01138     .WORD   1, 32           ;
0021  0021  0113C     .WORD   33, 33          ;
001E  001E  01140     .WORD   30, 30          ;
001E  001E  01144     .WORD   30, 30          ;
001E  001E  01148     .WORD   30, 30          ;
001E  001E  0114C     .WORD   30, 30          ;
001E  001E  01150     .WORD   30, 30          ;
001E  001E  01154     .WORD   30, 30          ;
0003  001E  01158     .WORD   30, 3           ;
001F  0000  0115C     .WORD   0, 31           ;
0020  0000  01160     .WORD   0, 32           ;
0021  0000  01164     .WORD   0, 33           ;
0022  0000  01168     .WORD   0, 34           ;
001E  001E  0116C     .WORD   30, 30          ;
001E  001E  01170     .WORD   30, 30          ;
001E  001E  01174     .WORD   30, 30          ;
001E  001E  01178     .WORD   30, 30          ;
001E  001E  0117C     .WORD   30, 30          ;
001E  001E  01180     .WORD   30, 30          ;
001E  001E  01184     .WORD   30, 30          ;
001E  001E  01188     .WORD   30, 30          ;
001D  001D  0118C     .WORD   29, 29          ;
001D  001D  01190     .WORD   29, 29          ;
001D  001D  01194     .WORD   29, 29          ;
001E  001E  01198     .WORD   30, 30          ;
001E  001E  0119C     .WORD   30, 30          ;
001E  001E  011A0     .WORD   30, 30          ;
001E  001E  011A4     .WORD   30, 30          ;
001E  001E  011A8     .WORD   30, 30          ;
```

```
001E  001E  011AC     .WORD   30, 30          ;
001E  001E  011B0     .WORD   30, 30          ;
001E  001E  011B4     .WORD   30, 30          ;
001E  001E  011B8     .WORD   30, 30          ;
001F  0001  011BC     .WORD   1, 31           ;
0020  0001  011C0     .WORD   1, 32           ;
0020  0001  011C4     .WORD   1, 32           ;
0020  0001  011C8     .WORD   1, 32           ;
0021  0021  011CC     .WORD   33, 33          ;
001E  001E  011D0     .WORD   30, 30          ;
001E  001E  011D4     .WORD   30, 30          ;
001E  001E  011D8     .WORD   30, 30          ;
001E  001E  011DC     .WORD   30, 30          ;
001E  001E  011E0     .WORD   30, 30          ;
001E  001E  011E4     .WORD   30, 30          ;
0003  001E  011E8     .WORD   30, 3           ;
001F  0000  011EC     .WORD   0, 31           ;
0020  0000  011F0     .WORD   0, 32           ;
0021  0000  011F4     .WORD   0, 33           ;
0022  0000  011F8     .WORD   0, 34           ;
001E  001E  011FC     .WORD   30, 30          ;
001E  001E  01200     .WORD   30, 30          ;
001E  001E  01204     .WORD   30, 30          ;
001E  001E  01208     .WORD   30, 30          ;
001E  001E  0120C     .WORD   30, 30          ;
001E  001E  01210     .WORD   30, 30          ;
001E  001E  01214     .WORD   30, 30          ;
001E  001E  01218     .WORD   30, 30          ;
001D  001D  0121C     .WORD   29, 29          ;
001D  001D  01220     .WORD   29, 29          ;
001D  001D  01224     .WORD   29, 29          ;
001E  001E  01228     .WORD   30, 30          ;
001E  001E  0122C     .WORD   30, 30          ;
001E  001E  01230     .WORD   30, 30          ;
001E  001E  01234     .WORD   30, 30          ;
001E  001E  01238     .WORD   30, 30          ;
001E  001E  0123C     .WORD   30, 30          ;
001E  001E  01240     .WORD   30, 30          ;
001E  001E  01244     .WORD   30, 30          ;
001E  001E  01248     .WORD   30, 30          ;
001F  0001  0124C     .WORD   1, 31           ;
0020  0001  01250     .WORD   1, 32           ;
0020  0001  01254     .WORD   1, 32           ;
0020  0001  01258     .WORD   1, 32           ;
0021  0021  0125C     .WORD   33, 33          ;
001E  001E  01260     .WORD   30, 30          ;
001E  001E  01264     .WORD   30, 30          ;
001E  001E  01268     .WORD   30, 30          ;
001E  001E  0126C     .WORD   30, 30          ;
001E  001E  01270     .WORD   30, 30          ;
001E  001E  01274     .WORD   30, 30          ;
0003  001E  01278     .WORD   30, 3           ;
001F  0000  0127C     .WORD   0, 31           ;
0020  0000  01280     .WORD   0, 32           ;
0021  0000  01284     .WORD   0, 33           ;
```

```
0022  0000  01288     .WORD   0, 34           ;
001E  001E  0128C     .WORD   30, 30          ;
001E  001E  01290     .WORD   30, 30          ;
001E  001E  01294     .WORD   30, 30          ;
001E  001E  01298     .WORD   30, 30          ;
001E  001E  0129C     .WORD   30, 30          ;
001E  001E  012A0     .WORD   30, 30          ;
001E  001E  012A4     .WORD   30, 30          ;
001E  001E  012A8     .WORD   30, 30          ;
001D  001D  012AC     .WORD   29, 29          ;
001D  001D  012B0     .WORD   29, 29          ;
001D  001D  012B4     .WORD   29, 29          ;
001E  001E  012B8     .WORD   30, 30          ;
001E  001E  012BC     .WORD   30, 30          ;
001E  001E  012C0     .WORD   30, 30          ;
001E  001E  012C4     .WORD   30, 30          ;
001E  001E  012C8     .WORD   30, 30          ;
001E  001E  012CC     .WORD   30, 30          ;
001E  001E  012D0     .WORD   30, 30          ;
001E  001E  012D4     .WORD   30, 30          ;
001E  001E  012D8     .WORD   30, 30          ;
001F  0001  012DC     .WORD   1, 31           ;
0020  0001  012E0     .WORD   1, 32           ;
0020  0001  012E4     .WORD   1, 32           ;
0020  0001  012E8     .WORD   1, 32           ;
0021  0021  012EC     .WORD   33, 33          ;
001E  001E  012F0     .WORD   30, 30          ;
001E  001E  012F4     .WORD   30, 30          ;
001E  001E  012F8     .WORD   30, 30          ;
001E  001E  012FC     .WORD   30, 30          ;
001E  001E  01300     .WORD   30, 30          ;
001E  001E  01304     .WORD   30, 30          ;
0003  001E  01308     .WORD   30, 3           ;
001F  0000  0130C     .WORD   0, 31           ;
0020  0000  01310     .WORD   0, 32           ;
0021  0000  01314     .WORD   0, 33           ;
0022  0000  01318     .WORD   0, 34           ;
001E  001E  0131C     .WORD   30, 30          ;
001E  001E  01320     .WORD   30, 30          ;
001E  001E  01324     .WORD   30, 30          ;
001E  001E  01328     .WORD   30, 30          ;
001E  001E  0132C     .WORD   30, 30          ;
001E  001E  01330     .WORD   30, 30          ;
001E  001E  01334     .WORD   30, 30          ;
001E  001E  01338     .WORD   30, 30          ;
001D  001D  0133C     .WORD   29, 29          ;
001D  001D  01340     .WORD   29, 29          ;
001D  001D  01344     .WORD   29, 29          ;
001E  001E  01348     .WORD   30, 30          ;
001E  001E  0134C     .WORD   30, 30          ;
001E  001E  01350     .WORD   30, 30          ;
001E  001E  01354     .WORD   30, 30          ;
001E  001E  01358     .WORD   30, 30          ;
001E  001E  0135C     .WORD   30, 30          ;
001E  001E  01360     .WORD   30, 30          ;
```

```
001E  001E  01364     .WORD   30, 30          ;
001E  001E  01368     .WORD   30, 30          ;
001F  0001  0136C     .WORD   1, 31           ;
0020  0001  01370     .WORD   1, 32           ;
0020  0001  01374     .WORD   1, 32           ;
0020  0001  01378     .WORD   1, 32           ;
0021  0021  0137C     .WORD   33, 33          ;
001E  001E  01380     .WORD   30, 30          ;
001E  001E  01384     .WORD   30, 30          ;
001E  001E  01388     .WORD   30, 30          ;
001E  001E  0138C     .WORD   30, 30          ;
001E  001E  01390     .WORD   30, 30          ;
001E  001E  01394     .WORD   30, 30          ;
0003  001E  01398     .WORD   30, 3           ;
001F  0000  0139C     .WORD   0, 31           ;
0020  0000  013A0     .WORD   0, 32           ;
0021  0000  013A4     .WORD   0, 33           ;
0022  0000  013A8     .WORD   0, 34           ;
001E  001E  013AC     .WORD   30, 30          ;
001E  001E  013B0     .WORD   30, 30          ;
001E  001E  013B4     .WORD   30, 30          ;
001E  001E  013B8     .WORD   30, 30          ;
001E  001E  013BC     .WORD   30, 30          ;
001E  001E  013C0     .WORD   30, 30          ;
001E  001E  013C4     .WORD   30, 30          ;
001E  001E  013C8     .WORD   30, 30          ;
001D  001D  013CC     .WORD   29, 29          ;
001D  001D  013D0     .WORD   29, 29          ;
001D  001D  013D4     .WORD   29, 29          ;
001E  001E  013D8     .WORD   30, 30          ;
001E  001E  013DC     .WORD   30, 30          ;
001E  001E  013E0     .WORD   30, 30          ;
001E  001E  013E4     .WORD   30, 30          ;
001E  001E  013E8     .WORD   30, 30          ;
001E  001E  013EC     .WORD   30, 30          ;
001E  001E  013F0     .WORD   30, 30          ;
001E  001E  013F4     .WORD   30, 30          ;
001E  001E  013F8     .WORD   30, 30          ;
001F  0001  013FC     .WORD   1, 31           ;
0020  0001  01400     .WORD   1, 32           ;
0020  0001  01404     .WORD   1, 32           ;
0020  0001  01408     .WORD   1, 32           ;
0021  0021  0140C     .WORD   33, 33          ;
001E  001E  01410     .WORD   30, 30          ;
001E  001E  01414     .WORD   30, 30          ;
001E  001E  01418     .WORD   30, 30          ;
001E  001E  0141C     .WORD   30, 30          ;
001E  001E  01420     .WORD   30, 30          ;
001E  001E  01424     .WORD   30, 30          ;
0003  001E  01428     .WORD   30, 3           ;
001F  0000  0142C     .WORD   0, 31           ;
0020  0000  01430     .WORD   0, 32           ;
0021  0000  01434     .WORD   0, 33           ;
0022  0000  01438     .WORD   0, 34           ;
001E  001E  0143C     .WORD   30, 30          ;
```

```
001E  001E  01440     .WORD   30, 30          ;
001E  001E  01444     .WORD   30, 30          ;
001E  001E  01448     .WORD   30, 30          ;
001E  001E  0144C     .WORD   30, 30          ;
001E  001E  01450     .WORD   30, 30          ;
001E  001E  01454     .WORD   30, 30          ;
001E  001E  01458     .WORD   30, 30          ;
001D  001D  0145C     .WORD   29, 29          ;
001D  001D  01460     .WORD   29, 29          ;
001D  001D  01464     .WORD   29, 29          ;
001E  001E  01468     .WORD   30, 30          ;
001E  001E  0146C     .WORD   30, 30          ;
001E  001E  01470     .WORD   30, 30          ;
001E  001E  01474     .WORD   30, 30          ;
001E  001E  01478     .WORD   30, 30          ;
001E  001E  0147C     .WORD   30, 30          ;
001E  001E  01480     .WORD   30, 30          ;
001E  001E  01484     .WORD   30, 30          ;
001E  001E  01488     .WORD   30, 30          ;
001F  0001  0148C     .WORD   1, 31           ;
0020  0001  01490     .WORD   1, 32           ;
0020  0001  01494     .WORD   1, 32           ;
0020  0001  01498     .WORD   1, 32           ;
0021  0021  0149C     .WORD   33, 33          ;
001E  001E  014A0     .WORD   30, 30          ;
001E  001E  014A4     .WORD   30, 30          ;
001E  001E  014A8     .WORD   30, 30          ;
001E  001E  014AC     .WORD   30, 30          ;
001E  001E  014B0     .WORD   30, 30          ;
001E  001E  014B4     .WORD   30, 30          ;
0003  001E  014B8     .WORD   30, 3           ;
001F  0000  014BC     .WORD   0, 31           ;
0020  0000  014C0     .WORD   0, 32           ;
0021  0000  014C4     .WORD   0, 33           ;
0022  0000  014C8     .WORD   0, 34           ;
001E  001E  014CC     .WORD   30, 30          ;
001E  001E  014D0     .WORD   30, 30          ;
001E  001E  014D4     .WORD   30, 30          ;
001E  001E  014D8     .WORD   30, 30          ;
001E  001E  014DC     .WORD   30, 30          ;
001E  001E  014E0     .WORD   30, 30          ;
001E  001E  014E4     .WORD   30, 30          ;
001E  001E  014E8     .WORD   30, 30          ;
001E  001E  014EC     .WORD   30, 30          ;
001E  001E  014F0     .WORD   30, 30          ;
001D  001D  014F4     .WORD   29, 29          ;
001E  001E  014F8     .WORD   30, 30          ;
001E  001E  014FC     .WORD   30, 30          ;
001E  001E  01500     .WORD   30, 30          ;
001E  001E  01504     .WORD   30, 30          ;
001E  001E  01508     .WORD   30, 30          ;
001E  001E  0150C     .WORD   30, 30          ;
001E  001E  01510     .WORD   30, 30          ;
001E  001E  01514     .WORD   30, 30          ;
001E  001E  01518     .WORD   30, 30          ;
```

```
      001E  001E  0151C     .WORD   30, 30          ;
      001E  001E  01520     .WORD   30, 30          ;
      001E  001E  01524     .WORD   30, 30          ;
      001E  001E  01528     .WORD   30, 30          ;
      0021  0021  0152C     .WORD   33, 33          ;
      001E  001E  01530     .WORD   30, 30          ;
      001E  001E  01534     .WORD   30, 30          ;
      001E  001E  01538     .WORD   30, 30          ;
      001E  001E  0153C     .WORD   30, 30          ;
      001E  001E  01540     .WORD   30, 30          ;
      001E  001E  01544     .WORD   30, 30          ;
      0003  001E  01548     .WORD   30, 3           ;
      001F  0000  0154C     .WORD   0, 31           ;
      0020  0000  01550     .WORD   0, 32           ;
      0021  0000  01554     .WORD   0, 33           ;
      0022  0000  01558     .WORD   0, 34           ;
      001E  001E  0155C     .WORD   30, 30          ;

          DS$ENDPASS==        65552
          DS$GPHARD==         65560
          DS$ABORT==          65568
          DS$SUMMARY==        65576
          DS$BGNSUB==         65584
          DS$ENDSUB==         65592
          DS$CKLOOP==         65600
          DS$INLOOP==         65608
          DS$ESCAPE==         65616
          DS$BREAK==          65624
          DS$WAITMS==         65632
          DS$WAITUS==         65640
          DS$CANWAIT==        65648
          DS$CNTRLC==         65656
          DS$ASKDATA==        65664
          DS$ASKVLD==         65672
          DS$ASKADR==         65680
          DS$ASKLGCL==        65688
          DS$ASKSTR==         65696
          DS$BRANCH==         65704
          DS$CVTREG==         65712
          DS$PARSE==          65720
          DS$ERRSYS==         65728
          DS$ERRDEV==         65736
          DS$ERRHARD==        65744
          DS$ERRSOFT==        65752
          DS$PRINTB==         65760
          DS$PRINTX==         65768
          DS$PRINTF==         65776
          DS$PRINTS==         65784
          DS$PRINTSIG==       65792
          DS$ERRPREP==        65800
          DS$SETPRIEXV==      65808
          DS$MAPDBGBLOCK==    65816
          DS$GETBUF==         65824
          DS$RELBUF==         65832
          DS$MMON==           65872
```

```
          DS$MMOFF==        65880
          DS$SETVEC==        65888
          DS$CLRVEC==        65896
          DS$INITSCB==        65904
          DS$SETIPL==        65912
          DS$CHANNEL==        65920
          DS$SETMAP==        65928
          DS$SHOCHAN==        65936
          DS$LOAD==        65944
          DS$PROBE==        65952
          DS$ATTACH==        65960
          DS$HELP==        65968
          DS$FREEDBGSYM==        65976
          DS$LOADPCS==        65984
          DS$GETTERM==        65992
          DS$SERVICE_DISPATCHER==
                66000
          DS$MEMSIZE==        66008
          SYS$QIOW==        66048
          SYS$ALLOC==        66104
          SYS$ASCTIM==        66120
          SYS$ASSIGN==        66128
          SYS$BINTIM==        66136
          SYS$CANCEL==        66144
          SYS$CANTIM==        66152
          SYS$CLREF==        66200
          SYS$DALLOC==        66264
          SYS$DASSGN==        66272
          SYS$FAO==        66384
          SYS$FAOL==        66392
          SYS$GETTIM==        66424
          SYS$GTCHAN==        66432
          SYS$HIBER==        66432
          SYS$LKWSET==        66464
          SYS$NUMTIM==        66488
          SYS$QIO==        66504
          SYS$READEF==        66512
          SYS$SETAST==        66552
          SYS$SETEF==        66560
          SYS$SETIMR==        66592
          SYS$SETPRI==        66600
          SYS$SETPRT==        66608
          SYS$SETRWN==        66616
          SYS$ULKPAG==        66656
          SYS$ULWSET==        66664
          SYS$UNWIND==        66672
          SYS$WAITFR==        66680
          SYS$WFLAND==        66696
          SYS$WFLOR==        66704
          SYS$GETCHN==        66760
          SYS$GET==        66944
          SYS$READ==        66960
          SYS$CLOSE==        67000
          SYS$CONNECT==        67008
          SYS$DISCONNECT==        67024
```

```
          SYS$OPEN==       67080
          CRD$GW_STATE_TABLE==P.AAA
            .EXTRN  CRD$GET_STATE, CRD$GET_ACTION_ROUTINE
            .EXTRN  CRD$INIT_STATE_TABLE
            .EXTRN  CRD$TURING_MACHINE
            .EXTRN  CRD$PRINT_ACTION_ROUTINE

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0000 00000     .ENTRY  CRD$INIT_STATE_TABLE, Save nothing       ; 0620
       00000000'  EF  D4 00002     CLRL    CRD$GL_CURRENT_TYPECODE          ; 0645
       00000000'  EF       03  D0 00008     MOVL    #3, CRD$GL_CURRENT_STATE         ; 0646
       00000000'  EF       24  D0 0000F     MOVL    #36, CRD$GL_CURRENT_ACTION         ; 0647
        04 00016     RET                 ; 0650

; Routine Size:  23 bytes,    Routine Base:  CODE + 0000
```

```
;    0651  1 %SBTTL 'CRD$Get_State Routine'
;    0652  1
;    0653  1 GLOBAL ROUTINE CRD$Get_State =
;    0654  1
;    0655  1 !++
;    0656  1 ! Functional Description:
;    0657  1 !
;    0658  1 ! Get the current state number from the CRD$GW_State_Table.
;    0659  1 !
;    0660  1 ! Formal Input Parameters:
;    0661  1 !
;    0662  1 ! None
;    0663  1 !
;    0664  1 ! Formal Output Parameters:
;    0665  1 !
;    0666  1 ! None
;    0667  1 !
;    0668  1 ! Side Effects:
;    0669  1 !
;    0670  1 ! None
;    0671  1 !
;    0672  1 ! Completion Codes:
;    0673  1 !
;    0674  1 ! Returns the state number.
;    0675  1 !--
```

```
;    0676  2 BEGIN      ! Routine CRD$Get_State
;    0677  2
;    0678  3  RETURN (.CRD$GW_State_Table [.CRD$GL_Current_TypeCode, .CRD$GL_Current_State, CRD$K_New_State])
;    0679  3
;    0680  1 END;      ! Routine CRD$Get_State



          0000 00000     .ENTRY  CRD$GET_STATE, Save nothing          ; 0653
        50 00000000'  EF 00000090   8F  C5 00002     MULL3   #144, CRD$GL_CURRENT_TYPECODE, R0       ; 0678
        51 00000000'  EF  D0 0000E     MOVL    CRD$GL_CURRENT_STATE, R1         ;
        50    6041  DE 00015     MOVAL   (R0)[R1], R0             ;
        26 00000000'  EF  D1 00019     CMPL    CRD$GL_CURRENT_TYPECODE, #38         ;
        05  18 00020     BGEQ    1$               ;
        24       51  D1 00022     CMPL    R1, #36           ;
        05  19 00025     BLSS    2$               ;
        50       01  CE 00027 1$:   MNEGL   #1, R0           ;
        08  11 0002A     BRB     3$               ;
        50 00000000'EF40  9E 0002C 2$:   MOVAB   CRD$GW_STATE_TABLE[R0], R0         ;
        50 02    A0  3C 00034 3$:   MOVZWL  2(R0), R0           ;
      04 00038     RET                ; 0680

; Routine Size:  57 bytes,    Routine Base:  CODE + 0017
```

```
;    0681  1 %SBTTL 'CRD$Get_Action_Routine Routine'
;    0682  1
;    0683  1 GLOBAL ROUTINE CRD$Get_Action_Routine =
;    0684  1
;    0685  1 !++
;    0686  1 ! Functional Description:
;    0687  1 !
;    0688  1 ! Extract the current action routine number from the CRD$GW_State_Table and return this number.
;    0689  1 !
;    0690  1 ! Formal Input Parameters:
;    0691  1 !
;    0692  1 ! None
;    0693  1 !
;    0694  1 ! Formal Output Parameters:
;    0695  1 !
;    0696  1 ! None
;    0697  1 !
;    0698  1 ! Side Effects:
;    0699  1 !
;    0700  1 ! None
;    0701  1 !
;    0702  1 ! Completion Codes:
;    0703  1 !
;    0704  1 ! Returns the current action routine number.
;    0705  1 !--
```

```
;   0706  2 BEGIN     ! Routine CRD$Get_Action_Routine
;   0707  2
;   0708  3  RETURN (.CRD$GW_State_Table [.CRD$GL_Current_TypeCode, .CRD$GL_Current_State, CRD$K_Action_Routine])
;   0709  3
;   0710  1 END;      ! Routine CRD$Get_Action_Routine



          0000 00000     .ENTRY  CRD$GET_ACTION_ROUTINE, Save nothing       ; 0683
        50 00000000'  EF 00000090   8F  C5 00002    MULL3   #144, CRD$GL_CURRENT_TYPECODE, R0       ; 0708
        51 00000000'  EF  D0 0000E    MOVL    CRD$GL_CURRENT_STATE, R1        ;
        50    6041  DE 00015    MOVAL   (R0)[R1], R0            ;
        26 00000000'  EF  D1 00019    CMPL    CRD$GL_CURRENT_TYPECODE, #38        ;
        05  18 00020    BGEQ    1$              ;
        24       51  D1 00022    CMPL    R1, #36         ;
        05  19 00025    BLSS    2$              ;
        50       01  CE 00027 1$:   MNEGL   #1, R0          ;
        08  11 0002A    BRB     3$              ;
        50 00000000'EF40  9E 0002C 2$:   MOVAB   CRD$GW_STATE_TABLE[R0], R0        ;
        50       60  3C 00034 3$:   MOVZWL  (R0), R0            ;
      04 00037    RET                 ; 0710

; Routine Size:  56 bytes,    Routine Base:  CODE + 0050
```

```
;    0711  1 %SBTTL 'CRD$Turing_Machine Routine'
;    0712  1
;    0713  1 GLOBAL ROUTINE CRD$Turing_Machine (TypeCode, Length, Address) =
;    0714  1
;    0715  1 !++
;    0716  1 ! Functional Description:
;    0717  1 !
;    0718  1 ! This routine is the main decision-making routine for CRD-AutoTest. It simulates a finite-state automata
;    0719  1 ! and uses a state table to determine the next  state  of CRD. This state table is referenced using the
;    0720  1 ! typecode passed to it by the CRD$DS_Interface routine and the current 'state'. The initial state is
;    0721  1 ! CRD$K_State_Start (i.e., 0). The state table contains a new state number and an action routine number.
;    0722  1 ! The action routine number determines what action routine is called to perform some action based on the
;    0723  1 ! typecode (e.g., loading a DS command into the CLI data buffer, adjusting internal-CRD pointers, etc.).
;    0724  1 !
;    0725  1 ! This routine is capable of being called recursively. In fact, if an action routine returns a
;    0726  1 ! CRD$K_Repeat_Turing status value, this routine will call itself recursively. This allows multiple actions
;    0727  1 ! to be taken on a single typecode.
;    0728  1 !
;    0729  1 ! Formal Input Parameters:
;    0730  1 !
;    0731  1 ! TypeCode: The DS typecode value.
;    0732  1 ! Length:   The length of the string being printed out by the Resident-DS, or the length of the
;    0733  1 !    buffer in which to put a string (e.g., the length of the CLI data buffer).
;    0734  1 ! Address:  The address of the string being printed out, or the address of where to put the string.
;    0735  1 !
;    0736  1 ! Formal Output Parameters:
;    0737  1 !
;    0738  1 ! None
;    0739  1 !
;    0740  1 ! Side Effects:
;    0741  1 !
;    0742  1 ! None
;    0743  1 !
;    0744  1 ! Completion Codes:
;    0745  1 !
;    0746  1 ! CRD$K_Success: Branch around what was about to be done in the Resident-DS.
;    0747  1 ! CRD$K_Failure: Perform the action that was about to happen in the Resident-DS.
;    0748  1 !--
```

```
;    0749  2 BEGIN      ! Routine CRD$Turing_Machine
;    0750  2
;    0751  2  OWN
;    0752  2   Recursion : INITIAL (0);
;    0753  2
;    0754  2  MACRO
; M  0755  2   Action_Routine_Case (AR_Name) [] =
; M  0756  2    [%NAME ('CRD$K_', AR_Name)]:
; M  0757  2     BEGIN
; M  0758  2     LOCAL
; M  0759  2      Status : VECTOR [2, WORD];
; M  0760  2
; M  0761  2     !+
; M  0762  2     ! Call the action routine. Each routine returns a multi-value status number.
; M  0763  2     ! One such number is CRD$K_Repeat_Turing. When a routine returns this status
; M  0764  2     ! value, it means that this Turing_Machine routine must be called recursively.
; M  0765  2     ! This allows multiple actions to be performed on a single typecode (such as
; M  0766  2     ! the DS_Prompt typecode). The Load_xxxx_Com routines return the length of the
; M  0767  2     ! loaded DS command in the high word of the returned status value, so check
; M  0768  2     ! for the actual status value in the low word. Return the full longword to the
; M  0769  2     ! DS_Interface routine. It will return value to the Resident-DS routine that
; M  0770  2     ! called it. So, this length will get passed back to the DS_SuperLine routine
; M  0771  2     ! in the SCRIPT module.
; M  0772  2     !-
; M  0773  2
; M  0774  2     Status = %NAME ('CRD$', AR_Name) (%REMAINING);
; M  0775  2
; M  0776  2     IF .Status [0] EQL CRD$K_Repeat_Turing THEN
; M  0777  2      BEGIN
; M  0778  2      Status = CRD$Turing_Machine (.TypeCode, .Length, .Address);
; M  0779  2      Recursion = .Recursion - 1;
; M  0780  2      RETURN (.Status)
; M  0781  2      END
; M  0782  2     ELSE
; M  0783  2      BEGIN
; M  0784  2      Recursion = .Recursion - 1;
; M  0785  2      RETURN (.Status)
; M  0786  2      END
; M  0787  2     END
;    0788  2   %;
;    0789  2
;    0790  2  $DS_Break;    ! Check for ^C's typed on the terminal
;    0791  2
;    0792  2  IF (.CRD$GB_CRD_Debug) THEN
;    0793  2   $CRD_Print ($ASCIC ('!/Turing_Machine (!UL): The current '), .Recursion);
;    0794  2
;    0795  2  Recursion = .Recursion + 1;
;    0796  2
;    0797  2  IF (.CRD$GB_CRD_Debug) THEN
;    0798  3   BEGIN
;    0799  3   CRD$Print_TypeCode_Name (.TypeCode);
; P  0800  3   $CRD_Print
; P  0801  3    (
; P  0802  3    $ASCIC ('  - State: !2UL, Action: !2UL, TypeCode: !2UL, Address: !XL(X), Length: !3UL!/'),
; P  0803  3    .CRD$GL_Current_State, .CRD$GL_Current_Action, .TypeCode, .Address, .Length
```

```
;    0804  3     );
;    0805  2    END;
;    0806  2
;    0807  2   CRD$GL_Current_TypeCode = .TypeCode;
;    0808  2
;    0809  2   DO
;    0810  3    BEGIN
;    0811  3    REGISTER
;    0812  3     State, Action;
;    0813  3
;    0814  3    !+
;    0815  3    ! Don't store directly into the current_state and current_action because they are used
;    0816  3    ! in the two routines. Keep getting a new state until an action routine other than
;    0817  3    ! Advance_State is found.
;    0818  3    !-
;    0819  3
;    0820  3    State  = CRD$Get_State ();
;    0821  3    Action = CRD$Get_Action_Routine ();
;    0822  3
;    0823  3    CRD$GL_Current_State  = .State;
;    0824  3    CRD$GL_Current_Action = .Action
;    0825  3    END
;    0826  2   UNTIL (.CRD$GL_Current_Action NEQ CRD$K_Advance_State);
;    0827  2
;    0828  2   IF (.CRD$GB_CRD_Debug) THEN
;    0829  2    $CRD_Print ($ASCIC ('  - Executing action routine: '));
;    0830  2
;    0831  2   IF (.CRD$GL_Current_Action EQL CRD$K_Do_Nothing) THEN
;    0832  3    BEGIN
;    0833  3    Recursion = .Recursion - 1;
;    0834  3    IF (.CRD$GB_CRD_Debug) THEN
;    0835  3     CRD$Print_Action_Routine (CRD$K_Do_Nothing);
;    0836  4    RETURN (CRD$K_Success)
;    0837  2    END;
;    0838  2
;    0839  2   IF (.CRD$GB_CRD_Debug) THEN
;    0840  2    CRD$Print_Action_Routine (.CRD$GL_Current_Action);
;    0841  2
;    0842  2   CASE .CRD$GL_Current_Action FROM 0 TO (CRD$K_Last_Action_Routine - 1) OF
;    0843  2    SET
;    0844  2    Action_Routine_Case (Level_4_Passed);
;    0845  2
;    0846  2    Action_Routine_Case (Load_AutoSizer,        .Length, .Address);
;    0847  2    Action_Routine_Case (AutoSizer_Set_Flags,       .Length, .Address);
;    0848  2    Action_Routine_Case (Start_AutoSizer,        .Length, .Address);
;    0849  2    Action_Routine_Case (Assign_PTable_Ptrs);
;    0850  2
;    0851  2    Action_Routine_Case (Load_General_Com,         .Length, .Address);
;    0852  2    Action_Routine_Case (Advance_General_Com);
;    0853  2    Action_Routine_Case (Done_General_Coms);
;    0854  2
;    0855  2    Action_Routine_Case (Set_Up_Diagnostic);
;    0856  2
;    0857  2    Action_Routine_Case (Load_Load_Com,        .Length, .Address);
;    0858  2    Action_Routine_Case (Load_Set_Flags_Com,       .Length, .Address);
```

```
;    0859  2    Action_Routine_Case (Load_Set_Event_Com,      .Length, .Address);
;    0860  2    Action_Routine_Case (Load_Select_Com,       .Length, .Address);
;    0861  2    Action_Routine_Case (Load_Start_Com,       .Length, .Address);
;    0862  2
;    0863  2    Action_Routine_Case (Advance_Diagnostic);
;    0864  2    Action_Routine_Case (Exit_CRD);
;    0865  2
;    0866  2    Action_Routine_Case (AutoSizer_Error,   .TypeCode, .Length, .Address);
;    0867  2    Action_Routine_Case (Internal_Error,   .TypeCode, .Length, .Address);
;    0868  2    Action_Routine_Case (Load_Error);
;    0869  2    Action_Routine_Case (Start_Error);
;    0870  2    Action_Routine_Case (Diagnostic_Error,   .TypeCode, .Length, .Address);
;    0871  2    Action_Routine_Case (Preparation_Error,  .TypeCode, .Length, .Address);
;    0872  2
;    0873  2    [CRD$K_Do_Nothing,
;    0874  2     CRD$K_Advance_State,
;    0875  2     CRD$K_Start,
;    0876  2     CRD$K_Dummy_3,
;    0877  2     CRD$K_Dummy_4,
;    0878  2     CRD$K_Dummy_5,
;    0879  2     CRD$K_Dummy_6,
;    0880  2     CRD$K_Dummy_7,
;    0881  2     CRD$K_Dummy_8,
;    0882  2     CRD$K_Dummy_9,
;    0883  2     CRD$K_Dummy_10,
;    0884  2     CRD$K_Dummy_11,
;    0885  2     CRD$K_Dummy_12,
;    0886  2     CRD$K_Dummy_13]:
;    0887  3     BEGIN
;    0888  3     CRD$Internal_Error (CRD$K_Action_EQL_Do_Nothing, .Length, .Address);
;    0889  3       ! Pass length and address so that we have 3 parameters
;    0890  2     END;
;    0891  2
;    0892  2    [OUTRANGE]:
;    0893  3     BEGIN
;    0894  3     CRD$Internal_Error (CRD$K_Action_Range_Error, .Length, .Address);
;    0895  3       ! Pass length and address so that we have 3 parameters
;    0896  2     END;
;    0897  2    TES;
;    0898  2
;    0899  3   RETURN (CRD$K_Failure)
;    0900  1 END;      ! Routine CRD$Turing_Machine


            .PSECT  WORK,NOEXE,2

         00000000  0000C RECURSION:
            .LONG   0           ;

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

69  68  63  61  4D  5F  67  6E  69  72  75  54  2F  21  24  01560 P.AAB:  .ASCII  \$!/Turing_Machine (!UL): The current \     ;
63  20  65  68  54  20  3A  29  4C  55  21  28  20  65  6E  0156F             ;
    20  74  6E  65  72  72  75  0157E             ;
```

```
55  32  21  20  3A  65  74  61  74  53  20  2D  20  20  4E  01585 P.AAC:  .ASCII  \N  - State: !2UL, Action: !2UL, TypeCode\  ;
4C  55  32  21  20  3A  6E  6F  69  74  63  41  20  2C  4C  01594             ;
      65  64  6F  43  65  70  79  54  20  2C  015A3                 ;
73  73  65  72  64  64  41  20  2C  4C  55  32  21  20  3A  015AD    .ASCII  \: !2UL, Address: !XL(X), Length: !3UL!/\   ;
74  67  6E  65  4C  20  2C  29  58  28  4C  58  21  20  3A  015BC             ;
    2F  21  4C  55  33  21  20  3A  68  015CB                 ;
20  67  6E  69  74  75  63  65  78  45  20  2D  20  20  1E  015D4 P.AAD:  .ASCII  <30>\  - Executing action routine: \      ;
3A  65  6E  69  74  75  6F  72  20  6E  6F  69  74  63  61  015E3             ;
        20  015F2               ;

            .EXTRN  CRD$GB_CRD_DEBUG
            .EXTRN  CRD$PRINT, CRD$PRINT_TYPECODE_NAME
            .EXTRN  CRD$LEVEL_4_PASSED
            .EXTRN  CRD$LOAD_AUTOSIZER
            .EXTRN  CRD$AUTOSIZER_SET_FLAGS
            .EXTRN  CRD$START_AUTOSIZER
            .EXTRN  CRD$ASSIGN_PTABLE_PTRS
            .EXTRN  CRD$LOAD_GENERAL_COM
            .EXTRN  CRD$ADVANCE_GENERAL_COM
            .EXTRN  CRD$DONE_GENERAL_COMS
            .EXTRN  CRD$SET_UP_DIAGNOSTIC
            .EXTRN  CRD$LOAD_LOAD_COM
            .EXTRN  CRD$LOAD_SET_FLAGS_COM
            .EXTRN  CRD$LOAD_SET_EVENT_COM
            .EXTRN  CRD$LOAD_SELECT_COM
            .EXTRN  CRD$LOAD_START_COM
            .EXTRN  CRD$ADVANCE_DIAGNOSTIC
            .EXTRN  CRD$EXIT_CRD, CRD$AUTOSIZER_ERROR
            .EXTRN  CRD$INTERNAL_ERROR
            .EXTRN  CRD$LOAD_ERROR, CRD$START_ERROR
            .EXTRN  CRD$DIAGNOSTIC_ERROR
            .EXTRN  CRD$PREPARATION_ERROR

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              000C 00000     .ENTRY  CRD$TURING_MACHINE, Save R2,R3       ; 0713
 00000000G  9F       00  FB 00002     CALLS   #0, a#DS$BREAK              ; 0788
     17 00000000G   EF  E9 00009     BLBC    CRD$GB_CRD_DEBUG, 1$        ; 0792
 00000000'  EF   DD 00010     PUSHL   RECURSION           ; 0793
 00000000'  EF   9F 00016     PUSHAB  P.AAB           ;
     01  DD 0001C     PUSHL   #1          ;
     1A  DD 0001E     PUSHL   #26         ;
 00000000G  FF       04  FB 00020     CALLS   #4, aCRD$PRINT          ;
 00000000'  EF   D6 00027 1$:    INCL    RECURSION           ; 0795
     2B 00000000G   EF  E9 0002D     BLBC    CRD$GB_CRD_DEBUG, 2$        ; 0797
04   AC  DD 00034     PUSHL   TYPECODE            ; 0799
 00000000G  EF       01  FB 00037     CALLS   #1, CRD$PRINT_TYPECODE_NAME     ;
08   AC  DD 0003E     PUSHL   LENGTH          ; 0804
0C   AC  DD 00041     PUSHL   ADDRESS         ;
04   AC  DD 00044     PUSHL   TYPECODE            ;
     7E 00000000'   EF  7D 00047     MOVQ    CRD$GL_CURRENT_STATE, -(SP)     ;
 00000000'  EF   9F 0004E     PUSHAB  P.AAC           ;
     01  DD 00054     PUSHL   #1          ;
     1A  DD 00056     PUSHL   #26         ;
 00000000G  FF       08  FB 00058     CALLS   #8, aCRD$PRINT          ;
```

```
                53 04   AC  D0 0005F 2$:   MOVL    TYPECODE, R3            ; 0807
    00000000'   EF      53  D0 00063     MOVL    R3, CRD$GL_CURRENT_TYPECODE        ;
  FF20   CF      00   FB 0006A 3$:   CALLS   #0, CRD$GET_STATE           ; 0820
          52      50   D0 0006F     MOVL    R0, STATE                ;
  FF51   CF      00   FB 00072     CALLS   #0, CRD$GET_ACTION_ROUTINE        ; 0821
    00000000'   EF      52  D0 00077     MOVL    STATE, CRD$GL_CURRENT_STATE        ; 0823
    00000000'   EF      50  D0 0007E     MOVL    ACTION, CRD$GL_CURRENT_ACTION        ; 0824
        01 00000000'  EF  D1 00085     CMPL    CRD$GL_CURRENT_ACTION, #1         ; 0826
        DC  13 0008C     BEQL    3$              ;
        11 00000000G  EF  E9 0008E     BLBC    CRD$GB_CRD_DEBUG, 4$          ; 0828
    00000000'   EF  9F 00095     PUSHAB  P.AAD             ; 0829
        01  DD 0009B     PUSHL   #1              ;
        1A  DD 0009D     PUSHL   #26             ;
    00000000G   FF      03  FB 0009F     CALLS   #3, @CRD$PRINT          ;
    00000000'   EF  D5 000A6 4$:   TSTL    CRD$GL_CURRENT_ACTION          ; 0831
        1A  12 000AC     BNEQ    6$              ;
    00000000'   EF  D7 000AE     DECL    RECURSION             ; 0833
        09 00000000G  EF  E9 000B4     BLBC    CRD$GB_CRD_DEBUG, 5$          ; 0834
        7E  D4 000BB     CLRL    -(SP)             ; 0835
    00000000V   EF      01  FB 000BD     CALLS   #1, CRD$PRINT_ACTION_ROUTINE        ;
        50      01  D0 000C4 5$:   MOVL    #1, R0            ; 0836
     04 000C7     RET             ;
        0D 00000000G  EF  E9 000C8 6$:   BLBC    CRD$GB_CRD_DEBUG, 7$          ; 0839
    00000000'   EF  DD 000CF     PUSHL   CRD$GL_CURRENT_ACTION          ; 0840
    00000000V   EF      01  FB 000D5     CALLS   #1, CRD$PRINT_ACTION_ROUTINE        ;
        23        00 00000000'  EF  CF 000DC 7$:   CASEL   CRD$GL_CURRENT_ACTION, #0, #35       ; 0842
017C        017C      017C    017C    000E4 8$:   .WORD   35$-8$,-          ;
0071        0064      017C    0052    000EC     35$-8$,-          ;
017C        008B      017C    007E    000F4     35$-8$,-          ;
00A1        0094      017C    00B3    000FC     35$-8$,-          ;
00C9        00BC      017C    00AA    00104     9$-8$,-         ;
017C        00F2      00E4    00D6    0010C     35$-8$,-          ;
010A        017C      0100    017C    00114     12$-8$,-          ;
0134        0124      0114    017C    0011C     13$-8$,-          ;
017C        0158      0148    013E    00124     14$-8$,-          ;
          35$-8$,-          ;
          15$-8$,-          ;
          35$ 8$,-          ;
          19$ 8$,-          ;
          35$ 8$,-          ;
          16$ 8$,           ;
          17$ 8$,           ;
          18$ 8$,           ;
          35$ 8$,           ;
          20$ 8$,           ;
          21$ 8$,           ;
          22$ 8$,           ;
          23$ 8$,           ;
          24$ 8$,           ;
          35$ 8$,           ;
          35$ 8$,           ;
          25$ 8$,           ;
          35$ 8$,           ;
          26$ 8$,           ;
          35$ 8$,           ;
```

```
                27$ 8$,             ;
                28$-8$,-            ;
                29$-8$,-            ;
                30$-8$,-            ;
                31$-8$,-            ;
                32$-8$,-            ;
                35$-8$              ;
            7E 08    AC  7D 0012C     MOVQ    LENGTH, -(SP)          ; 0894
            7E       02  CE 00130     MNEGL   #2, -(SP)          ;
          0131  31 00133     BRW     36$             ;
      00000000G  EF       00  FB 00136 9$:   CALLS   #0, CRD$LEVEL_4_PASSED        ; 0844
            02       50  B1 0013D 10$:   CMPW    STATUS, #2          ;
            03  13 00140     BEQL    11$             ;
          0114  31 00142     BRW     34$             ;
          0106  31 00145 11$:   BRW     33$             ;
            7E 08    AC  7D 00148 12$:   MOVQ    LENGTH, -(SP)          ; 0846
      00000000G  EF       02  FB 0014C     CALLS   #2, CRD$LOAD_AUTOSIZER        ;
            E8  11 00153     BRB     10$             ;
            7E 08    AC  7D 00155 13$:   MOVQ    LENGTH, -(SP)          ; 0847
      00000000G  EF       02  FB 00159     CALLS   #2, CRD$AUTOSIZER_SET_FLAGS       ;
            DB  11 00160     BRB     10$             ;
            7E 08    AC  7D 00162 14$:   MOVQ    LENGTH, -(SP)          ; 0848
      00000000G  EF       02  FB 00166     CALLS   #2, CRD$START_AUTOSIZER       ;
            CE  11 0016D     BRB     10$             ;
      00000000G  EF       00  FB 0016F 15$:   CALLS   #0, CRD$ASSIGN_PTABLE_PTRS        ; 0849
            C5  11 00176     BRB     10$             ;
            7E 08    AC  7D 00178 16$:   MOVQ    LENGTH, -(SP)          ; 0851
      00000000G  EF       02  FB 0017C     CALLS   #2, CRD$LOAD_GENERAL_COM        ;
            B8  11 00183     BRB     10$             ;
      00000000G  EF       00  FB 00185 17$:   CALLS   #0, CRD$ADVANCE_GENERAL_COM        ; 0852
            AF  11 0018C     BRB     10$             ;
      00000000G  EF       00  FB 0018E 18$:   CALLS   #0, CRD$DONE_GENERAL_COMS        ; 0853
            A6  11 00195     BRB     10$             ;
      00000000G  EF       00  FB 00197 19$:   CALLS   #0, CRD$SET_UP_DIAGNOSTIC        ; 0855
            9D  11 0019E     BRB     10$             ;
            7E 08    AC  7D 001A0 20$:   MOVQ    LENGTH, -(SP)          ; 0857
      00000000G  EF       02  FB 001A4     CALLS   #2, CRD$LOAD_LOAD_COM         ;
            90  11 001AB     BRB     10$             ;
            7E 08    AC  7D 001AD 21$:   MOVQ    LENGTH, -(SP)          ; 0858
      00000000G  EF       02  FB 001B1     CALLS   #2, CRD$LOAD_SET_FLAGS_COM        ;
            83  11 001B8     BRB     10$             ;
            7E 08    AC  7D 001BA 22$:   MOVQ    LENGTH, -(SP)          ; 0859
      00000000G  EF       02  FB 001BE     CALLS   #2, CRD$LOAD_SET_EVENT_COM        ;
          FF75  31 001C5     BRW     10$             ;
            7E 08    AC  7D 001C8 23$:   MOVQ    LENGTH, -(SP)          ; 0860
      00000000G  EF       02  FB 001CC     CALLS   #2, CRD$LOAD_SELECT_COM       ;
          FF67  31 001D3     BRW     10$             ;
            7E 08    AC  7D 001D6 24$:   MOVQ    LENGTH, -(SP)          ; 0861
      00000000G  EF       02  FB 001DA     CALLS   #2, CRD$LOAD_START_COM        ;
          FF59  31 001E1     BRW     10$             ;
      00000000G  EF       00  FB 001E4 25$:   CALLS   #0, CRD$ADVANCE_DIAGNOSTIC        ; 0863
          FF4F  31 001EB     BRW     10$             ;
      00000000G  EF       00  FB 001EE 26$:   CALLS   #0, CRD$EXIT_CRD          ; 0864
          FF45  31 001F5     BRW     10$             ;
            7E 08    AC  7D 001F8 27$:   MOVQ    LENGTH, -(SP)          ; 0866
```

```
                53  DD 001FC     PUSHL   R3                  ;
            00000000G  EF       03  FB 001FE     CALLS   #3, CRD$AUTOSIZER_ERROR        ;
              FF35  31 00205     BRW     10$                 ;
                7E 08   AC  7D 00208 28$:   MOVQ    LENGTH, -(SP)           ; 0867
                53  DD 0020C     PUSHL   R3                  ;
            00000000G  EF       03  FB 0020E     CALLS   #3, CRD$INTERNAL_ERROR        ;
              FF25  31 00215     BRW     10$                 ;
            00000000G  EF       00  FB 00218 29$:   CALLS   #0, CRD$LOAD_ERROR          ; 0868
              FF1B  31 0021F     BRW     10$                 ;
            00000000G  EF       00  FB 00222 30$:   CALLS   #0, CRD$START_ERROR          ; 0869
              FF11  31 00229     BRW     10$                 ;
                7E 08   AC  7D 0022C 31$:   MOVQ    LENGTH, -(SP)           ; 0870
                53  DD 00230     PUSHL   R3                  ;
            00000000G  EF       03  FB 00232     CALLS   #3, CRD$DIAGNOSTIC_ERROR        ;
              FF01  31 00239     BRW     10$                 ;
                7E 08   AC  7D 0023C 32$:   MOVQ    LENGTH, -(SP)           ; 0871
                53  DD 00240     PUSHL   R3                  ;
            00000000G  EF       03  FB 00242     CALLS   #3, CRD$PREPARATION_ERROR       ;
                02      50  B1 00249     CMPW    STATUS, #2          ;
                0B  12 0024C     BNEQ    34$                 ;
                7E 08   AC  7D 0024E 33$:   MOVQ    LENGTH, -(SP)           ;
                53  DD 00252     PUSHL   R3                  ;
           FDA7   CF       03  FB 00254     CALLS   #3, CRD$TURING_MACHINE          ;
            00000000'  EF  D7 00259 34$:   DECL    RECURSION           ;
             04 0025F     RET                 ;
                7E 08   AC  7D 00260 35$:   MOVQ    LENGTH, -(SP)           ; 0888
                7E      03  CE 00264     MNEGL   #3, -(SP)           ;
            00000000G  EF       03  FB 00267 36$:   CALLS   #3, CRD$INTERNAL_ERROR        ;
                50  D4 0026E     CLRL    R0                  ; 0899
             04 00270     RET                 ; 0900

; Routine Size:  625 bytes,    Routine Base:  CODE + 0088
```

```
;    0901  1 %SBTTL 'CRD$Print_Action_Routine Routine'
;    0902  1
;    0903  1 GLOBAL ROUTINE CRD$Print_Action_Routine (Action_Routine) : NOVALUE =
;    0904  1
;    0905  1 !++
;    0906  1 ! Functional Description:
;    0907  1 !
;    0908  1 ! This routine prints a message saying what the Action_Routine is.
;    0909  1 !
;    0910  1 ! Formal Input Parameters:
;    0911  1 !
;    0912  1 ! Action_Routine : An action routine constant.
;    0913  1 !
;    0914  1 ! Formal Output Parameters:
;    0915  1 !
;    0916  1 ! None
;    0917  1 !
;    0918  1 ! Side Effects:
;    0919  1 !
;    0920  1 ! None
;    0921  1 !
;    0922  1 ! Completion Codes:
;    0923  1 !
;    0924  1 ! None
;    0925  1 !--
```

```
;   0926  2 BEGIN     ! Routine CRD$Print_Action_Routine
;   0927  2  MACRO
; M 0928  2   Action_Routine_Case (Action_Routine_String) =
; M 0929  2    [%NAME ('CRD$K_', Action_Routine_String)]:
; M 0930  2     $CRD_Print ($ASCIC ('(!UL) - ', Action_Routine_String, '!/!/'), .Action_Routine)
;   0931  2   %;
;   0932  2
;   0933  2  CASE .Action_Routine FROM 0 TO (CRD$K_Last_Action_Routine - 1) OF
;   0934  2   SET
;   0935  2   Action_Routine_Case ('Do_Nothing');
;   0936  2   Action_Routine_Case ('Advance_State');
;   0937  2   Action_Routine_Case ('Dummy_3');
;   0938  2
;   0939  2   Action_Routine_Case ('Start');
;   0940  2   Action_Routine_Case ('Level_4_Passed');
;   0941  2   Action_Routine_Case ('Dummy_4');
;   0942  2
;   0943  2   Action_Routine_Case ('Load_AutoSizer');
;   0944  2   Action_Routine_Case ('AutoSizer_Set_Flags');
;   0945  2   Action_Routine_Case ('Start_AutoSizer');
;   0946  2   Action_Routine_Case ('Dummy_5');
;   0947  2   Action_Routine_Case ('Assign_PTable_Ptrs');
;   0948  2   Action_Routine_Case ('Dummy_6');
;   0949  2
;   0950  2   Action_Routine_Case ('Load_General_Com');
;   0951  2   Action_Routine_Case ('Advance_General_Com');
;   0952  2   Action_Routine_Case ('Done_General_Coms');
;   0953  2   Action_Routine_Case ('Dummy_7');
;   0954  2
;   0955  2   Action_Routine_Case ('Set_Up_Diagnostic');
;   0956  2   Action_Routine_Case ('Dummy_8');
;   0957  2
;   0958  2   Action_Routine_Case ('Load_Load_Com');
;   0959  2   Action_Routine_Case ('Load_Set_Flags_Com');
;   0960  2   Action_Routine_Case ('Load_Set_Event_Com');
;   0961  2   Action_Routine_Case ('Load_Select_Com');
;   0962  2   Action_Routine_Case ('Load_Start_Com');
;   0963  2   Action_Routine_Case ('Dummy_9');
;   0964  2   Action_Routine_Case ('Dummy_10');
;   0965  2
;   0966  2   Action_Routine_Case ('Advance_Diagnostic');
;   0967  2   Action_Routine_Case ('Dummy_11');
;   0968  2   Action_Routine_Case ('Exit_CRD');
;   0969  2   Action_Routine_Case ('Dummy_12');
;   0970  2
;   0971  2   Action_Routine_Case ('AutoSizer_Error');
;   0972  2   Action_Routine_Case ('Internal_Error');
;   0973  2   Action_Routine_Case ('Load_Error');
;   0974  2   Action_Routine_Case ('Start_Error');
;   0975  2   Action_Routine_Case ('Diagnostic_Error');
;   0976  2   Action_Routine_Case ('Preparation_Error');
;   0977  2   Action_Routine_Case ('Dummy_13');
;   0978  2
;   0979  2   [OUTRANGE]:
; P 0980  2    $CRD_Print ($ASCIC (' Print_Action_Routine - Illegal action routine number - !UL!/'),
```

```
;   0981  2      .Action_Routine);
;   0982  2    TES;
;   0983  1 END;      ! Routine CRD$Print_Action_Routine



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

74  6F  4E  5F  6F  44  20  2D  20  29  4C  55  21  28  16  015F3 P.AAE:  .ASCII  <22>\(!UL) - Do_Nothing!/!/\        ;
        2F  21  2F  21  67  6E  69  68  01602                          ;
63  6E  61  76  64  41  20  2D  20  29  4C  55  21  28  19  0160A P.AAF:  .ASCII  <25>\(!UL) - Advance_State!/!/\       ;
  2F  21  2F  21  65  74  61  74  53  5F  65  01619                      ;
5F  79  6D  6D  75  44  20  2D  20  29  4C  55  21  28  13  01624 P.AAG:  .ASCII  <19>\(!UL) - Dummy_3!/!/\        ;
    2F  21  2F  21  33  01633                  ;
21  74  72  61  74  53  20  2D  20  29  4C  55  21  28  11  01638 P.AAH:  .ASCII  <17>\(!UL) - Start!/!/\        ;
      2F  21  2F  01647                      ;
5F  6C  65  76  65  4C  20  2D  20  29  4C  55  21  28  1A  0164A P.AAI:  .ASCII  <26>\(!UL) - Level_4_Passed!/!/\       ;
    2F  21  2F  21  64  65  73  73  61  50  5F  34  01659                  ;
5F  79  6D  6D  75  44  20  2D  20  29  4C  55  21  28  13  01665 P.AAJ:  .ASCII  <19>\(!UL) - Dummy_4!/!/\        ;
    2F  21  2F  21  34  01674                  ;
41  5F  64  61  6F  4C  20  2D  20  29  4C  55  21  28  1A  01679 P.AAK:  .ASCII  <26>\(!UL) - Load_AutoSizer!/!/\       ;
    2F  21  2F  21  72  65  7A  69  53  6F  74  75  01688                  ;
69  53  6F  74  75  41  20  2D  20  29  4C  55  21  28  1F  01694 P.AAL:  .ASCII  <31>\(!UL) - AutoSizer_Set_Flags!/!/\       ;
2F  21  73  67  61  6C  46  5F  74  65  53  5F  72  65  7A  016A3              ;
        2F  21  016B2                      ;
5F  74  72  61  74  53  20  2D  20  29  4C  55  21  28  1B  016B4 P.AAM:  .ASCII  <27>\(!UL) - Start_AutoSizer!/!/\       ;
 2F  21  2F  21  72  65  7A  69  53  6F  74  75  41  016C3                  ;
5F  79  6D  6D  75  44  20  2D  20  29  4C  55  21  28  13  016D0 P.AAN:  .ASCII  <19>\(!UL) - Dummy_5!/!/\        ;
    2F  21  2F  21  35  016DF                  ;
6E  67  69  73  73  41  20  2D  20  29  4C  55  21  28  1E  016E4 P.AAO:  .ASCII  <30>\(!UL) - Assign_PTable_Ptrs!/!/\       ;
21  2F  21  73  72  74  50  5F  65  6C  62  61  54  50  5F  016F3              ;
        2F  01702                      ;
5F  79  6D  6D  75  44  20  2D  20  29  4C  55  21  28  13  01703 P.AAP:  .ASCII  <19>\(!UL) - Dummy_6!/!/\        ;
    2F  21  2F  21  36  01712                  ;
47  5F  64  61  6F  4C  20  2D  20  29  4C  55  21  28  1C  01717 P.AAQ:  .ASCII  <28>\(!UL) - Load_General_Com!/!/\       ;
    2F  21  2F  21  6D  6F  43  5F  6C  61  72  65  6E  65  01726              ;
63  6E  61  76  64  41  20  2D  20  29  4C  55  21  28  1F  01734 P.AAR:  .ASCII  <31>\(!UL) - Advance_General_Com!/!/\       ;
2F  21  6D  6F  43  5F  6C  61  72  65  6E  65  47  5F  65  01743              ;
          2F  21  01752                    ;
47  5F  65  6E  6F  44  20  2D  20  29  4C  55  21  28  1D  01754 P.AAS:  .ASCII  <29>\(!UL) - Done_General_Coms!/!/\       ;
2F  21  2F  21  73  6D  6F  43  5F  6C  61  72  65  6E  65  01763              ;
5F  79  6D  6D  75  44  20  2D  20  29  4C  55  21  28  13  01772 P.AAT:  .ASCII  <19>\(!UL) - Dummy_7!/!/\        ;
    2F  21  2F  21  37  01781                  ;
70  55  5F  74  65  53  20  2D  20  29  4C  55  21  28  1D  01786 P.AAU:  .ASCII  <29>\(!UL) - Set_Up_Diagnostic!/!/\       ;
2F  21  2F  21  63  69  74  73  6F  6E  67  61  69  44  5F  01795              ;
5F  79  6D  6D  75  44  20  2D  20  29  4C  55  21  28  13  017A4 P.AAV:  .ASCII  <19>\(!UL) - Dummy_8!/!/\        ;
    2F  21  2F  21  38  017B3                  ;
4C  5F  64  61  6F  4C  20  2D  20  29  4C  55  21  28  19  017B8 P.AAW:  .ASCII  <25>\(!UL) - Load_Load_Com!/!/\       ;
  2F  21  2F  21  6D  6F  43  5F  64  61  6F  017C7                        ;
53  5F  64  61  6F  4C  20  2D  20  29  4C  55  21  28  1E  017D2 P.AAX:  .ASCII  <30>\(!UL) - Load_Set_Flags_Com!/!/\       ;
21  2F  21  6D  6F  43  5F  73  67  61  6C  46  5F  74  65  017E1              ;
        2F  017F0                      ;
53  5F  64  61  6F  4C  20  2D  20  29  4C  55  21  28  1E  017F1 P.AAY:  .ASCII  <30>\(!UL) - Load_Set_Event_Com!/!/\       ;
21  2F  21  6D  6F  43  5F  74  6E  65  76  45  5F  74  65  01800              ;
      2F  0180F                    ;
```

```
53 5F 64 61 6F 4C 20 2D 20 29 4C 55 21 28 1B  01810 P.AAZ:  .ASCII  <27>\(!UL) - Load_Select_Com!/!/\       ;
 2F 21 2F 21 6D 6F 43 5F 74 63 65 6C 65 0181F                        ;
53 5F 64 61 6F 4C 20 2D 20 29 4C 55 21 28 1A  0182C P.ABA:  .ASCII  <26>\(!UL) - Load_Start_Com!/!/\       ;
    2F 21 2F 21 6D 6F 43 5F 74 72 61 74 0183B                        ;
5F 79 6D 6D 75 44 20 2D 20 29 4C 55 21 28 13  01847 P.ABB:  .ASCII  <19>\(!UL) - Dummy_9!/!/\        ;
    2F 21 2F 21 39 01856                ;
5F 79 6D 6D 75 44 20 2D 20 29 4C 55 21 28 14  0185B P.ABC:  .ASCII  <20>\(!UL) - Dummy_10!/!/\        ;
       2F 21 2F 21 30 31 0186A                  ;
63 6E 61 76 64 41 20 2D 20 29 4C 55 21 28 1E  01870 P.ABD:  .ASCII  <30>\(!UL) - Advance_Diagnostic!/!/\       ;
21 2F 21 63 69 74 73 6F 6E 67 61 69 44 5F 65  0187F                        ;
      2F 0188E                  ;
5F 79 6D 6D 75 44 20 2D 20 29 4C 55 21 28 14  0188F P.ABE:  .ASCII  <20>\(!UL) - Dummy_11!/!/\        ;
       2F 21 2F 21 31 31 0189E                  ;
43 5F 74 69 78 45 20 2D 20 29 4C 55 21 28 14  018A4 P.ABF:  .ASCII  <20>\(!UL) - Exit_CRD!/!/\        ;
       2F 21 2F 21 44 52 018B3                  ;
5F 79 6D 6D 75 44 20 2D 20 29 4C 55 21 28 14  018B9 P.ABG:  .ASCII  <20>\(!UL) - Dummy_12!/!/\        ;
       2F 21 2F 21 32 31 018C8                  ;
69 53 6F 74 75 41 20 2D 20 29 4C 55 21 28 1B  018CE P.ABH:  .ASCII  <27>\(!UL) - AutoSizer_Error!/!/\       ;
 2F 21 2F 21 72 6F 72 72 45 5F 72 65 7A 018DD                        ;
6E 72 65 74 6E 49 20 2D 20 29 4C 55 21 28 1A  018EA P.ABI:  .ASCII  <26>\(!UL) - Internal_Error!/!/\       ;
    2F 21 2F 21 72 6F 72 72 45 5F 6C 61 018F9                        ;
45 5F 64 61 6F 4C 20 2D 20 29 4C 55 21 28 16  01905 P.ABJ:  .ASCII  <22>\(!UL) - Load_Error!/!/\        ;
      2F 21 2F 21 72 6F 72 72 01914                        ;
5F 74 72 61 74 53 20 2D 20 29 4C 55 21 28 17  0191C P.ABK:  .ASCII  <23>\(!UL) - Start_Error!/!/\       ;
   2F 21 2F 21 72 6F 72 72 45 0192B                        ;
6F 6E 67 61 69 44 20 2D 20 29 4C 55 21 28 1C  01934 P.ABL:  .ASCII  <28>\(!UL) - Diagnostic_Error!/!/\       ;
    2F 21 2F 21 72 6F 72 72 45 5F 63 69 74 73 01943                        ;
72 61 70 65 72 50 20 2D 20 29 4C 55 21 28 1D  01951 P.ABM:  .ASCII  <29>\(!UL) - Preparation_Error!/!/\       ;
2F 21 2F 21 72 6F 72 72 45 5F 6E 6F 69 74 61 01960                        ;
5F 79 6D 6D 75 44 20 2D 20 29 4C 55 21 28 14  0196F P.ABN:  .ASCII  <20>\(!UL) - Dummy_13!/!/\        ;
       2F 21 2F 21 33 31 0197E                  ;
5F 6E 6F 69 74 63 41 5F 74 6E 69 72 50 20 3D  01984 P.ABO:  .ASCII  \= Print_Action_Routine - Illegal action \  ;
67 65 6C 6C 49 20 2D 20 65 6E 69 74 75 6F 52  01993                        ;
      20 6E 6F 69 74 63 61 20 6C 61 019A2                  ;
20 72 65 62 6D 75 6E 20 65 6E 69 74 75 6F 72  019AC     .ASCII  \routine number = !UL!/\        ;
   2F 21 4C 55 21 20 3D 019BB                ;


          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            0000 00000     .ENTRY  CRD$PRINT_ACTION_ROUTINE, Save nothing      ; 0903
          51 00000000G   EF  D0 00002     MOVL    CRD$PRINT, R1           ; 0981
          50 04    AC  D0 00009     MOVL    ACTION_ROUTINE, R0          ; 0933
          23         00       50  CF 0000D     CASEL   R0, #0, #35             ;
  0074          0069       005E     0053      00011 1$:    .WORD   2$-1$,-          ;
  00A0          0095       008A     007F      00019     3$-1$,-           ;
  00CC          00C1       00B6     00AB      00021     4$-1$,-           ;
  00E2          00D7       00F8     0103      00029     5$-1$,-           ;
  0124          0119       010E     00ED      00031     6$-1$,            ;
  0150          0145       013A     012F      00039     7$-1$,            ;
  0178          016E       0164     015A      00041     8$-1$,            ;
  01A0          0196       018C     0182      00049     9$ 1$,            ;
  01C8          01BE       01B4     01AA      00051     10$ 1$,            ;
           11$-1$,-          ;
```

```
            12$-1$,-          ;
            13$-1$,-          ;
            18$-1$,-          ;
            17$-1$,-          ;
            14$ 1$,-          ;
            15$-1$,-          ;
            16$-1$,-          ;
            19$ 1$,-          ;
            20$ 1$,-          ;
            21$-1$,-          ;
            22$ 1$,-          ;
            23$-1$,           ;
            24$ 1$,           ;
            25$-1$,           ;
            26$ 1$,-          ;
            27$ 1$,-          ;
            28$ 1$,-          ;
            29$ 1$,           ;
            30$ 1$,-          ;
            31$-1$,-          ;
            32$-1$,-          ;
            33$-1$,-          ;
            34$-1$,-          ;
            35$-1$,-          ;
            36$-1$,-          ;
            37$-1$          ;
        50  DD 00059     PUSHL   R0              ; 0981
00000000'   EF  9F 0005B     PUSHAB  P.ABO              ;
  017D  31  00061     BRW     38$             ;
    50  DD  00064 2$:   PUSHL   R0              ; 0935
00000000'   EF  9F 00066     PUSHAB  P.AAE              ;
  0172  31  0006C     BRW     38$             ;
    50  DD  0006F 3$:   PUSHL   R0              ; 0936
00000000'   EF  9F 00071     PUSHAB  P.AAF              ;
  0167  31  00077     BRW     38$             ;
    50  DD  0007A 4$:   PUSHL   R0              ; 0937
00000000'   EF  9F 0007C     PUSHAB  P.AAG              ;
  015C  31  00082     BRW     38$             ;
    50  DD  00085 5$:   PUSHL   R0              ; 0939
00000000'   EF  9F 00087     PUSHAB  P.AAH              ;
  0151  31  0008D     BRW     38$             ;
    50  DD  00090 6$:   PUSHL   R0              ; 0940
00000000'   EF  9F 00092     PUSHAB  P.AAI              ;
  0146  31  00098     BRW     38$             ;
    50  DD  0009B 7$:   PUSHL   R0              ; 0941
00000000'   EF  9F 0009D     PUSHAB  P.AAJ              ;
  013B  31  000A3     BRW     38$             ;
    50  DD  000A6 8$:   PUSHL   R0              ; 0943
00000000'   EF  9F 000AB     PUSHAB  P.AAK              ;
  0130  31  000AE     BRW     38$             ;
    50  DD  000B1 9$:   PUSHL   R0              ; 0944
00000000'   EF  9F 000B3     PUSHAB  P.AAL              ;
  0125  31  000B9     BRW     38$             ;
    50  DD  000BC 10$:   PUSHL   R0              ; 0945
00000000'   EF  9F 000BE     PUSHAB  P.AAM              ;
```

```
          011A  31 000C4     BRW     38$             ;
            50  DD 000C7 11$:   PUSHL   R0              ; 0946
      00000000'  EF  9F 000C9     PUSHAB  P.AAN               ;
            010F  31 000CF     BRW     38$             ;
            50  DD 000D2 12$:   PUSHL   R0              ; 0947
      00000000'  EF  9F 000D4     PUSHAB  P.AAO               ;
            0104  31 000DA     BRW     38$             ;
            50  DD 000DD 13$:   PUSHL   R0              ; 0948
      00000000'  EF  9F 000DF     PUSHAB  P.AAP               ;
            00F9  31 000E5     BRW     38$             ;
            50  DD 000E8 14$:   PUSHL   R0              ; 0950
      00000000'  EF  9F 000EA     PUSHAB  P.AAQ               ;
            00EE  31 000F0     BRW     38$             ;
            50  DD 000F3 15$:   PUSHL   R0              ; 0951
      00000000'  EF  9F 000F5     PUSHAB  P.AAR               ;
            00E3  31 000FB     BRW     38$             ;
            50  DD 000FE 16$:   PUSHL   R0              ; 0952
      00000000'  EF  9F 00100     PUSHAB  P.AAS               ;
            00D8  31 00106     BRW     38$             ;
            50  DD 00109 17$:   PUSHL   R0              ; 0953
      00000000'  EF  9F 0010B     PUSHAB  P.AAT               ;
            00CD  31 00111     BRW     38$             ;
            50  DD 00114 18$:   PUSHL   R0              ; 0955
      00000000'  EF  9F 00116     PUSHAB  P.AAU               ;
            00C2  31 0011C     BRW     38$             ;
            50  DD 0011F 19$:   PUSHL   R0              ; 0956
      00000000'  EF  9F 00121     PUSHAB  P.AAV               ;
            00B7  31 00127     BRW     38$             ;
            50  DD 0012A 20$:   PUSHL   R0              ; 0958
      00000000'  EF  9F 0012C     PUSHAB  P.AAW               ;
            00AC  31 00132     BRW     38$             ;
            50  DD 00135 21$:   PUSHL   R0              ; 0959
      00000000'  EF  9F 00137     PUSHAB  P.AAX               ;
            00A1  31 0013D     BRW     38$             ;
            50  DD 00140 22$:   PUSHL   R0              ; 0960
      00000000'  EF  9F 00142     PUSHAB  P.AAY               ;
            0096  31 00148     BRW     38$             ;
            50  DD 0014B 23$:   PUSHL   R0              ; 0961
      00000000'  EF  9F 0014D     PUSHAB  P.AAZ               ;
            008B  31 00153     BRW     38$             ;
            50  DD 00156 24$:   PUSHL   R0              ; 0962
      00000000'  EF  9F 00158     PUSHAB  P.ABA               ;
            0080  31 0015E     BRW     38$             ;
            50  DD 00161 25$:   PUSHL   R0              ; 0963
      00000000'  EF  9F 00163     PUSHAB  P.ABB               ;
            76  11 00169     BRB     38$             ;
            50  DD 0016B 26$:   PUSHL   R0              ; 0964
      00000000'  EF  9F 0016D     PUSHAB  P.ABC               ;
            6C  11 00173     BRB     38$             ;
            50  DD 00175 27$:   PUSHL   R0              ; 0966
      00000000'  EF  9F 00177     PUSHAB  P.ABD               ;
            62  11 0017D     BRB     38$             ;
            50  DD 0017F 28$:   PUSHL   R0              ; 0967
      00000000'  EF  9F 00181     PUSHAB  P.ABE               ;
            58  11 00187     BRB     38$             ;
```

```
            50  DD 00189 29$:   PUSHL   R0              ; 0968
      00000000'  EF  9F 0018B     PUSHAB  P.ABF               ;
            4E  11 00191     BRB     38$             ;
            50  DD 00193 30$:   PUSHL   R0              ; 0969
      00000000'  EF  9F 00195     PUSHAB  P.ABG               ;
            44  11 0019B     BRB     38$             ;
            50  DD 0019D 31$:   PUSHL   R0              ; 0971
      00000000'  EF  9F 0019F     PUSHAB  P.ABH               ;
            3A  11 001A5     BRB     38$             ;
            50  DD 001A7 32$:   PUSHL   R0              ; 0972
      00000000'  EF  9F 001A9     PUSHAB  P.ABI               ;
            30  11 001AF     BRB     38$             ;
            50  DD 001B1 33$:   PUSHL   R0              ; 0973
      00000000'  EF  9F 001B3     PUSHAB  P.ABJ               ;
            26  11 001B9     BRB     38$             ;
            50  DD 001BB 34$:   PUSHL   R0              ; 0974
      00000000'  EF  9F 001BD     PUSHAB  P.ABK               ;
            1C  11 001C3     BRB     38$             ;
            50  DD 001C5 35$:   PUSHL   R0              ; 0975
      00000000'  EF  9F 001C7     PUSHAB  P.ABL               ;
            12  11 001CD     BRB     38$             ;
            50  DD 001CF 36$:   PUSHL   R0              ; 0976
      00000000'  EF  9F 001D1     PUSHAB  P.ABM               ;
            08  11 001D7     BRB     38$             ;
            50  DD 001D9 37$:   PUSHL   R0              ; 0977
      00000000'  EF  9F 001DB     PUSHAB  P.ABN               ;
            01  DD 001E1 38$:   PUSHL   #1              ;
            1A  DD 001E3     PUSHL   #26             ;
            61       04  FB 001E5     CALLS   #4, (R1)            ;
         04 001E8     RET                 ; 0983

; Routine Size:  489 bytes,    Routine Base:  CODE + 02F9
```

```
;   0984  1 END     ! End of CRDSTATE module
;   0985  0 ELUDOM
```

```
;           PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  WORK           16  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  DATA         6594  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE         1250  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  . ABS  .          0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$ASCIC    Unbound          930
    Macro    Lib02   793     802     829     935     936     937     939     940     941     943
             944     945     946     947     948     950     951     952     953     955
             956     958     959     960     961     962     963     964     966     967
             968     969     971     972     973     974     975     976     977     980
$CRD_LITERALS   Macro    Lib03    85
$CRD_PRINT   Unbound          930
    Macro    Lib03   793     800     829     935     936     937     939     940     941     943
             944     945     946     947     948     950     951     952     953     955
             956     958     959     960     961     962     963     964     966     967
             968     969     971     972     973     974     975     976     977     980
$DS_BREAK   Macro    Lib02   790
$DS_DSSDEF   Macro    Lib02    88
$DS_TYPEDEF    Macro    Lib02    84     793     804     829     935     936     937     939     940     941
             943     944     945     946     947     948     950     951     952     953
             955     956     958     959     960     961     962     963     964     966
             967     968     969     971     972     973     974     975     976     977
             981
$ENTRY$     Unbound          140     143     145     147     149     166     168     222     257     266
             281     284     293     302     311     320     328     336     347     362
             371     380     385     394     399     408     417     427     442     457
             462     467     476     479     488     497     500     515     530     545
             560     575     590     605     614
    Comptime   132   222     257     266     281     284     293     302     311     320     328
             336     347     362     371     380     385     394     399     408     417
             427     442     457     462     467     476     479     488     497     500
             515     530     545     560     575     590     605     614
$EQULST     Macro    Lib04    84      85     793     804     829     935     936     937     939     940
             941     943     944     945     946     947     948     950     951     952
             953     955     956     958     959     960     961     962     963     964
             966     967     968     969     971     972     973     974     975     976
             977     981
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
$PREV_COLUMN$   Unbound          183     186     195     201     222     257     266     281     293     302
              311     320     328     336     347     362     371     380     385     394
              399     408     417     427     442     457     462     467     476     488
              497     515     530     545     560     575     590     605     614
    Comptime   129   222     257     266     281     293     302     311     320     328     336
              347     362     371     380     385     394     399     408     417     427
              442     457     462     467     476     488     497     500     515     530
              545     560     575     590     605     614
$ROWS$     Unbound        142     147     222     257     266     281     284     293     302     311
              320     328     336     347     362     371     380     385     394     399
              408     417     427     442     457     462     467     476     479     488
              497     500     515     530     545     560     575     590     605     614
    Comptime   133   222     257     266     281     284     293     302     311     320     328
              336     347     362     371     380     385     394     399     408     417
              427     442     457     462     467     476     479     488     497     500
              515     530     545     560     575     590     605     614
$ROW_CNT$   Comptime   131
$TEMP$     Comptime   128
$WINDOW_SIZE$   Unbound          186     188     190     201     203     205     222     257     266     281
              293     302     311     320     328     336     347     362     371     380
              385     394     399     408     417     427     442     457     462     467
              476     488     497     515     530     545     560     575     590     605
              614
    Comptime   130   222     257     266     281     293     302     311     320     328     336
              347     362     371     380     385     394     399     408     417     427
              442     457     462     467     476     488     497     500     515     530
              545     560     575     590     605     614
ACTION     MacrForm   180   193
    Register   812   821=    824.
ACTION_ROUTINE   Unbound          930
    MacrForm   136   138
    MacrForm   155   156
    RoutForm   903   933.    935.    936.    937.    939.    940.    941.    943.    944.    945.
              946.    947.    948.    950.    951.    952.    953.    955.    956.    958.
              959.    960.    961.    962.    963.    964.    966.    967.    968.    969.
              971.    972.    973.    974.    975.    976.    977.    981.
ACTION_ROUTINE_CASE  Macro      755   844     846     847     848     849     851     852     853     855     957
              858     859     860     861     863     864     866     867     868     869
              870     871
    Macro      928   935     936     937     939     940     941     943     944     945     946
              947     948     950     951     952     953     955     956     958     959
              960     961     962     963     964     966     967     968     969     971
              972     973     974     975     976     977
ACTION_ROUTINE_STRING  MacrForm   928   929     930
ADDRESS    Unbound          778
    RoutForm   713   804.    844.    846.    847.    848.    849.    851.    852.    853.    855.
              857.    858.    859.    860.    861.    863.    864.    866.    867.    868.
              869.    870.    871.    888.    894.
AR_NAME    MacrForm   755   756     774
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     85
AUTO$K_EXIT_CONTROL_C  Literal     85
AUTO$K_EXIT_DUMMY_2  Literal     85
AUTO$K_EXIT_DUMMY_3  Literal     85
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------- ------- ----- ---------------
AUTO$K_EXIT_ERRORS_COMPLETE Literal     85
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     85
AUTO$K_EXIT_INTERNAL_ERROR Literal     85
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     85
AUTO$K_EXIT_LAST_REASON  Literal     85     85
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     85
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     85
AUTO$K_EXIT_NOERRORS_PREP Literal     85
COLUMN    MacrForm   180   186     195
CRD$ADVANCE_DIAGNOSTIC  ExtRout  Lib03   863c
CRD$ADVANCE_GENERAL_COM  ExtRout  Lib03   852c
CRD$ASSIGN_PTABLE_PTRS  ExtRout  Lib03   849c
CRD$AUTOSIZER_ERROR  ExtRout  Lib03   866c
CRD$AUTOSIZER_SET_FLAGS  ExtRout  Lib03   847c
CRD$DIAGNOSTIC_ERROR  ExtRout  Lib03   870c
CRD$DONE_GENERAL_COMS  ExtRout  Lib03   853c
CRD$EXIT_CRD   ExtRout  Lib03   864c
CRD$GB_CRD_DEBUG  External Lib03   792.    797.    828.    834.    839.
CRD$GET_ACTION_ROUTINE  GlobRout   683 Lib03e     95f    821c
CRD$GET_STATE   GlobRout   653 Lib03e     92f    820c
CRD$GL_CURRENT_ACTION  Global     124 Lib03e    647=    804.    824=    826.    831.    840.    842.
CRD$GL_CURRENT_STATE  Global     123 Lib03e    646=    678.    708.    804.    823=
CRD$GL_CURRENT_TYPECODE  Global     122 Lib03e    645=    678.    708.    807=
CRD$GW_STATE_TABLE  External Lib03
    GlobBind   214   678     678.    708     708.
CRD$INIT_STATE_TABLE  GlobRout   620 Lib03e     98f
CRD$INTERNAL_ERROR  ExtRout  Lib03   867c    888c    894c
CRD$K_ACTION_EQL_DO_NOTHING Literal     85   888
CRD$K_ACTION_RANGE_ERROR Literal     85   894
CRD$K_ACTION_ROUTINE  Literal  Lib03   708
CRD$K_ADVANCE_DIAGNOSTIC Literal     85   257     863     966
CRD$K_ADVANCE_GENERAL_COM Literal     85   257     852     951
CRD$K_ADVANCE_STATE  Literal     85   257     281     347     362     442     457     467     515     530     545
              560     575     590     605     826     874     936
CRD$K_ALLOCATION_ERROR  Literal     85
CRD$K_ASSIGN_PTABLE_PTRS Literal     85   257     849     947
CRD$K_AUTOSIZER_ERROR  Literal     85   257     266     281     293     302     311     320     328     336     347
              362     371     408     417     427     442     457     467     476     488
              497     515     530     545     560     575     590     605     614     866
              971
CRD$K_AUTOSIZER_SET_FLAGS Literal     85   257     847     944
CRD$K_BAD_ACTION_NUMBER  Literal     85
CRD$K_BAD_TYPECODE_ERROR Literal     85
CRD$K_BASE_SYSTEM_IDC  Literal     85
CRD$K_BASE_SYSTEM_LESI  Literal     85
CRD$K_BASE_SYSTEM_NULL  Literal     85
CRD$K_BASE_SYSTEM_UDA_0  Literal     85
CRD$K_BASE_SYSTEM_UDA_1  Literal     85
CRD$K_BASE_SYSTEM_UDA_2  Literal     85
CRD$K_BASE_SYSTEM_UDA_3  Literal     85
CRD$K_CRD_INITIALIZATION Literal     85
CRD$K_CREATE_HST  Literal     85
CRD$K_DATA_LENGTH_ERROR  Literal     85
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     85
CRD$K_DDB_BLINK   Literal      85
CRD$K_DDB_FLINK   Literal      85
CRD$K_DDB_LAST_ENTRY  Literal      85
CRD$K_DDB_MEMORY_SIZE  Literal      85
CRD$K_DDB_MENU_DEVICE_NUMB Literal      85
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal      85
CRD$K_DDB_MENU_SUPPORT_SIZE Literal      85
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal      85
CRD$K_DDB_PTABLE_ENTRY  Literal      85
CRD$K_DDB_STATUS  Literal      85
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal      85
CRD$K_DEVICE_NAME  Literal      85
CRD$K_DIAGNOSTIC_ERROR  Literal      85   266     293     302     311     320     347     362     371     408     417
               427     476     497     515     530     545     560     575     590     605
               614     870     975
CRD$K_DIAGNOSTIC_NAME  Literal      85
CRD$K_DONE_GENERAL_COMS  Literal      85   257     853     952
CRD$K_DO_NOTHING  Literal      85   222     266     281     293     302     311     320     328     336     362
               371     380     385     394     399     408     417     427     442     457
               462     476     488     497     500     515     530     545     560     575
               590     605     614     831     835     873     935
CRD$K_DUMMY_10   Literal      85   883     964
CRD$K_DUMMY_11   Literal      85   884     967
CRD$K_DUMMY_12   Literal      85   885     969
CRD$K_DUMMY_13   Literal      85   886     977
CRD$K_DUMMY_3   Literal      85   876     937
CRD$K_DUMMY_4   Literal      85   877     941
CRD$K_DUMMY_5   Literal      85   878     946
CRD$K_DUMMY_6   Literal      85   879     948
CRD$K_DUMMY_7   Literal      85   880     953
CRD$K_DUMMY_8   Literal      85   881     956
CRD$K_DUMMY_9   Literal      85   882     963
CRD$K_EXIT_CRD   Literal      85   257     864     968
CRD$K_FAILURE   Literal  Lib03   899
CRD$K_HOOK_POINT  Literal      85
CRD$K_HS_INTERNAL_ERROR  Literal      85
CRD$K_IDC_EVDLB   Literal      85
CRD$K_IDC_EVDLC   Literal      85
CRD$K_IDC_EVDLD   Literal      85
CRD$K_IDC_EVRAD_0  Literal      85
CRD$K_IDC_EVRAD_1  Literal      85
CRD$K_IDC_LAST_DIAGNOSTIC Literal      85
CRD$K_INTERNAL_ERROR  Literal      85   222     257     266     281     284     293     302     311     320     328
               336     347     362     371     380     385     394     399     408     417
               427     442     457     462     467     476     479     488     497     500
               515     530     545     560     575     590     605     614     867     972
CRD$K_LAST_ACTION_ROUTINE Literal      85   647     842     933
CRD$K_LAST_ENTRY  Literal      85
CRD$K_LAST_FUNCTION  Literal      85
CRD$K_LAST_HOOK_POINT  Literal      85
CRD$K_LAST_INTERNAL_ERROR Literal      85
CRD$K_LAST_TYPE   Literal      85
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_LESI_ENWCA  Literal     85
CRD$K_LESI_EVRMB_0  Literal     85
CRD$K_LESI_EVRMB_1  Literal     85
CRD$K_LESI_LAST_DIAGNOSTIC Literal     85
CRD$K_LEVEL_4_PASSED  Literal     85   257     844     940
CRD$K_LOAD_AUTOSIZER  Literal     85   257     846     943
CRD$K_LOAD_ERROR  Literal     85   257     868     973
CRD$K_LOAD_GENERAL_COM  Literal     85   257     851     950
CRD$K_LOAD_LOAD_COM  Literal     85   257     857     958
CRD$K_LOAD_SELECT_COM  Literal     85   257     860     961
CRD$K_LOAD_SET_EVENT_COM Literal     85   257     859     960
CRD$K_LOAD_SET_FLAGS_COM Literal     85   257     858     959
CRD$K_LOAD_START_COM  Literal     85   257     861     962
CRD$K_LOAD_SUP_FILE  Literal     85
CRD$K_MM_INTERNAL_ERROR  Literal     85
CRD$K_NEW_LINE   Literal     85
CRD$K_NEW_STATE   Literal  Lib03   678
CRD$K_NUMB_STATES  Unbound          140     145     170     201     222     257     266     281     284     293
                  302     311     320     328     336     347     362     371     380     385
                  394     399     408     417     427     442     457     462     467     476
                  479     488     497     500     515     530     545     560     575     590
                  605     614
    Literal  Lib03   222     257     266     281     284     293     302     311     320     328
                  336     347     362     371     380     385     394     399     408     417
                  427     442     457     462     467     476     479     488     497     500
                  515     530     545     560     575     590     605     614     617
CRD$K_NUMB_TYPECODES  Literal  Lib03   211     617
CRD$K_PREPARATION_ERROR  Literal     85   488     871     976
CRD$K_PREPARATION_ERROR_TEXT Literal     85
CRD$K_REPEAT_TURING  Unbound          776
    Literal  Lib03   844     846     847     848     849     851     852     853     855     857
                  858     859     860     861     863     864     866     867     868     869
                  870     871
CRD$K_RUNTIME   Literal     85
CRD$K_SAME_LINE   Literal     85
CRD$K_SET_EVENT_ARGS  Literal     85
CRD$K_SET_FLAGS_ARGS  Literal     85
CRD$K_SET_UP_DIAGNOSTIC  Literal     85   257     855     955
CRD$K_START   Literal     85   875     939
CRD$K_START_AUTOSIZER  Literal     85   257     848     945
CRD$K_START_ERROR  Literal     85   257     869     974
CRD$K_START_QUALIFIERS  Literal     85
CRD$K_STAR_LINE   Literal     85
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal     85   257
CRD$K_STATE_ADVANCE_GENERAL_COM Literal     85   257
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal     85   257
CRD$K_STATE_AUTOSIZER_ERROR Literal     85   257     266     281     293     302     311     320     328     336     347
                  362     371     408     417     427     442     457     467     476     488
                  497     515     530     545     560     575     590     605     614
CRD$K_STATE_AUTOSIZER_RUNNING Literal     85   222     257     266     281     293     302     311     320     328     336
                  347     362     371     380     385     394     399     408     417     427
                  442     457     462     467     476     488     497     515     530     545
                  560     575     590     605     614
```

```
Symbol                            Type  Defined  Referenced ...
--------------------------------- ----- -------- --------------
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal     85   257     281     347     362     442     457     515     530     545     560
                575     590     605
CRD$K_STATE_DIAGNOSTIC_ERROR Literal     85   222     257     266     281     293     302     311     320     328     336
                347     362     371     380     394     408     417     427     442     457
                476     488     497     515     530     545     560     575     590     605
                614
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal     85   222     257     266     281     293     302     311     320     347     362
                371     380     385     394     399     408     417     427     442     457
                462     467     476     488     497     515     530     545     560     575
                590     605     614
CRD$K_STATE_DONE_DIAGNOSTICS Literal     85   257
CRD$K_STATE_DONE_GENERAL_COMS Literal     85   257
CRD$K_STATE_DUMMY_1  Literal     85
CRD$K_STATE_DUMMY_10  Literal     85
CRD$K_STATE_DUMMY_12  Literal     85
CRD$K_STATE_DUMMY_13  Literal     85
CRD$K_STATE_DUMMY_2  Literal     85
CRD$K_STATE_DUMMY_3  Literal     85
CRD$K_STATE_DUMMY_4  Literal     85
CRD$K_STATE_DUMMY_6  Literal     85
CRD$K_STATE_DUMMY_7  Literal     85
CRD$K_STATE_DUMMY_8  Literal     85
CRD$K_STATE_EXIT_CRD  Literal     85   257
CRD$K_STATE_INTERNAL_ERROR Literal     85   222     257     266     281     284     293     302     311     320     328
                336     347     362     371     380     385     394     399     408     417
                427     442     457     462     467     476     479     488     497     500
                515     530     545     560     575     590     605     614
CRD$K_STATE_LAST_STATE  Literal     85
CRD$K_STATE_LEVEL_4_PASSED Literal     85   257
CRD$K_STATE_LOAD_AUTOSIZER Literal     85   257
CRD$K_STATE_LOAD_ERROR  Literal     85   222     257     266     281     293     302     311     320     328     336
                347     362     371     380     394     408     417     427     442     457
                476     488     497     515     530     545     560     575     590     605
                614
CRD$K_STATE_LOAD_GENERAL_COM Literal     85   257
CRD$K_STATE_LOAD_LOAD_COM Literal     85   257
CRD$K_STATE_LOAD_SELECT_COM Literal     85   257     281     347     362     442     457     515     530     545     560
                575     590     605
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     85   257     281     347     362     442     457     515     530     545     560
                575     590     605
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     85   257     281     347     362     442     457     515     530     545     560
                575     590     605
CRD$K_STATE_LOAD_START_COM Literal     85   257     281     347     362     442     457     515     530     545     560
                575     590     605
CRD$K_STATE_PREPARATION_ERROR Literal     85   222     257     266     281     293     302     311     320     328     336
                362     371     380     394     408     417     427     442     457     476
                488     497     515     530     545     560     575     590     605     614
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     85   257
CRD$K_STATE_START  Literal     85   222     257     266     281     293     302     311     320     328     336
                347     362     371     380     385     394     399     408     417     427
                442     457     462     467     476     488     497     500     515     530
                545     560     575     590     605     614     646
CRD$K STATE_START_AUTOSIZER Literal     85   257     281     347     362     442     457     515     530     545     560
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
                575     590     605
CRD$K_STATE_START_ERROR  Literal     85   222     257     266     281     293     302     311     320     328     336
                347     362     371     380     394     408     417     427     442     457
                467     476     488     497     515     530     545     560     575     590
                605     614
CRD$K_SUCCESS   Literal  Lib03   836
CRD$K_TEST_START_TEXT  Literal     85
CRD$K_TQ_INTERNAL_ERROR  Literal     85
CRD$K_TYPECODE   Literal     85
CRD$K_UDA_0_EVDLB  Literal     85
CRD$K_UDA_0_EVDLC  Literal     85
CRD$K_UDA_0_EVDLD  Literal     85
CRD$K_UDA_0_EVRLA_0  Literal     85
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     85
CRD$K_UDA_1_EVDLB  Literal     85
CRD$K_UDA_1_EVDLC  Literal     85
CRD$K_UDA_1_EVDLD  Literal     85
CRD$K_UDA_1_EVRLA_0  Literal     85
CRD$K_UDA_1_EVRLA_1  Literal     85
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     85
CRD$K_UDA_2_EVDLB  Literal     85
CRD$K_UDA_2_EVDLC  Literal     85
CRD$K_UDA_2_EVDLD  Literal     85
CRD$K_UDA_2_EVRLA_0  Literal     85
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     85
CRD$K_UDA_3_EVDLB  Literal     85
CRD$K_UDA_3_EVDLC  Literal     85
CRD$K_UDA_3_EVDLD  Literal     85
CRD$K_UDA_3_EVRLA_0  Literal     85
CRD$K_UDA_3_EVRLA_1  Literal     85
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     85
CRD$LEVEL_4_PASSED  ExtRout  Lib03   844c
CRD$LOAD_AUTOSIZER  ExtRout  Lib03   846c
CRD$LOAD_ERROR   ExtRout  Lib03   868c
CRD$LOAD_GENERAL_COM  ExtRout  Lib03   851c
CRD$LOAD_LOAD_COM  ExtRout  Lib03   857c
CRD$LOAD_SELECT_COM  ExtRout  Lib03   860c
CRD$LOAD_SET_EVENT_COM  ExtRout  Lib03   859c
CRD$LOAD_SET_FLAGS_COM  ExtRout  Lib03   858c
CRD$LOAD_START_COM  ExtRout  Lib03   861c
CRD$PREPARATION_ERROR  ExtRout  Lib03   871c
CRD$PRINT   External Lib03   793ac   804ac   829ac   935ac   936ac   937ac   939ac   940ac   941ac   943ac
              944ac   945ac   946ac   947ac   948ac   950ac   951ac   952ac   953ac   955ac
              956ac   958ac   959ac   960ac   961ac   962ac   963ac   964ac   966ac   967ac
              968ac   969ac   971ac   972ac   973ac   974ac   975ac   976ac   977ac   981ac
CRD$PRINT_ACTION_ROUTINE GlobRout   903 Lib03e    104f    835c    840c
CRD$PRINT_TYPECODE_NAME  ExtRout  Lib03   799c
CRD$SET_UP_DIAGNOSTIC  ExtRout  Lib03   855c
CRD$START_AUTOSIZER  ExtRout  Lib03   848c
CRD$START_ERROR   ExtRout  Lib03   869c
CRD$TURING_MACHINE  Unbound          778
    GlobRout   713 Lib03e    101f    844c    846c    847c    848c    849c    851c    852c    853c
              855c    857c    858c    859c    860c    861c    863c    864c    866c    867c
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
                 868c    869c    870c    871c
CRD_ASSERT     Macro    Lib03   211
DS$ABORT     GlobLit      88
DS$ASKADR     GlobLit      88
DS$ASKDATA     GlobLit      88
DS$ASKLGCL     GlobLit      88
DS$ASKSTR     GlobLit      88
DS$ASKVLD     GlobLit      88
DS$ATTACH     GlobLit      88
DS$BGNSUB     GlobLit      88
DS$BRANCH     GlobLit      88
DS$BREAK     GlobLit      88
    ExtRout      790    790c
DS$CANWAIT     GlobLit      88
DS$CHANNEL     GlobLit      88
DS$CKLOOP     GlobLit      88
DS$CLRVEC     GlobLit      88
DS$CNTRLC     GlobLit      88
DS$CVTREG     GlobLit      88
DS$ENDPASS     GlobLit      88
DS$ENDSUB     GlobLit      88
DS$ERRDEV     GlobLit      88
DS$ERRHARD     GlobLit      88
DS$ERRPREP     GlobLit      88
DS$ERRSOFT     GlobLit      88
DS$ERRSYS     GlobLit      88
DS$ESCAPE     GlobLit      88
DS$FREEDBGSYM     GlobLit      88
DS$GETBUF     GlobLit      88
DS$GETTERM     GlobLit      88
DS$GPHARD     GlobLit      88
DS$HELP     GlobLit      88
DS$INITSCB     GlobLit      88
DS$INLOOP     GlobLit      88
DS$K_PRINTB     Literal      84
    Literal      793
    Literal      804
    Literal      829
    Literal      935
    Literal      936
    Literal      937
    Literal      939
    Literal      940
    Literal      941
    Literal      943
    Literal      944
    Literal      945
    Literal      946
    Literal      947
    Literal      948
    Literal      950
    Literal      951
    Literal      952
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_PRINTF   Literal      84
    Literal     793    793
    Literal     804    804
    Literal     829    829
    Literal     935    935
    Literal     936    936
    Literal     937    937
    Literal     939    939
    Literal     940    940
    Literal     941    941
    Literal     943    943
    Literal     944    944
    Literal     945    945
    Literal     946    946
    Literal     947    947
    Literal     948    948
    Literal     950    950
    Literal     951    951
    Literal     952    952
    Literal     953    953
    Literal     955    955
    Literal     956    956
    Literal     958    958
    Literal     959    959
    Literal     960    960
    Literal     961    961
    Literal     962    962
    Literal     963    963
    Literal     964    964
    Literal     966    966
    Literal     967    967
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     968   968
    Literal     969   969
    Literal     971   971
    Literal     972   972
    Literal     973   973
    Literal     974   974
    Literal     975   975
    Literal     976   976
    Literal     977   977
    Literal     981   981
DS$K_PRINTI   Literal      84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_PRINTX   Literal      84
    Literal     793
```

Symbol	Type	Defined	Referenced ...
	Literal		804
	Literal		829
	Literal		935
	Literal		936
	Literal		937
	Literal		939
	Literal		940
	Literal		941
	Literal		943
	Literal		944
	Literal		945
	Literal		946
	Literal		947
	Literal		948
	Literal		950
	Literal		951
	Literal		952
	Literal		953
	Literal		955
	Literal		956
	Literal		958
	Literal		959
	Literal		960
	Literal		961
	Literal		962
	Literal		963
	Literal		964
	Literal		966
	Literal		967
	Literal		968
	Literal		969
	Literal		971
	Literal		972
	Literal		973
	Literal		974
	Literal		975
	Literal		976
	Literal		977
	Literal		981
DS$K_TYPE_ABORT_PROGRAM	Literal	84	
	Literal		793
	Literal		804
	Literal		829
	Literal		935
	Literal		936
	Literal		937
	Literal		939
	Literal		940
	Literal		941
	Literal		943
	Literal		944
	Literal		945
	Literal		946

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_ABORT_TEST  Literal      84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_COMMAND_ERR  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_COMMAND_OUT  Literal      84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_CRD_AUTOTEST  Literal      84
    Literal     793    793
    Literal     804    804
    Literal     829    829
    Literal     935    935
    Literal     936    936
    Literal     937    937
    Literal     939    939
    Literal     940    940
```

Symbol	Type	Defined	Referenced ...
Literal		941	941
Literal		943	943
Literal		944	944
Literal		945	945
Literal		946	946
Literal		947	947
Literal		948	948
Literal		950	950
Literal		951	951
Literal		952	952
Literal		953	953
Literal		955	955
Literal		956	956
Literal		958	958
Literal		959	959
Literal		960	960
Literal		961	961
Literal		962	962
Literal		963	963
Literal		964	964
Literal		966	966
Literal		967	967
Literal		968	968
Literal		969	969
Literal		971	971
Literal		972	972
Literal		973	973
Literal		974	974
Literal		975	975
Literal		976	976
Literal		977	977
Literal		981	981
DS$K_TYPE_DS_PROMPT	Literal	84	
Literal		793	
Literal		804	
Literal		829	
Literal		935	
Literal		936	
Literal		937	
Literal		939	
Literal		940	
Literal		941	
Literal		943	
Literal		944	
Literal		945	
Literal		946	
Literal		947	
Literal		948	
Literal		950	
Literal		951	
Literal		952	
Literal		953	
Literal		955	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_DS_START  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_ERRDEV  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_ERRHARD  Literal     84
    Literal     793
    Literal     804
    Literal     829
```

Symbol	Type	Defined	Referenced ...
	Literal		935
	Literal		936
	Literal		937
	Literal		939
	Literal		940
	Literal		941
	Literal		943
	Literal		944
	Literal		945
	Literal		946
	Literal		947
	Literal		948
	Literal		950
	Literal		951
	Literal		952
	Literal		953
	Literal		955
	Literal		956
	Literal		958
	Literal		959
	Literal		960
	Literal		961
	Literal		962
	Literal		963
	Literal		964
	Literal		966
	Literal		967
	Literal		968
	Literal		969
	Literal		971
	Literal		972
	Literal		973
	Literal		974
	Literal		975
	Literal		976
	Literal		977
	Literal		981
DS$K_TYPE_ERROR_BODY	Literal		84
	Literal		793
	Literal		804
	Literal		829
	Literal		935
	Literal		936
	Literal		937
	Literal		939
	Literal		940
	Literal		941
	Literal		943
	Literal		944
	Literal		945
	Literal		946
	Literal		947
	Literal		948

Symbol	Type	Defined	Referenced ...
	Literal	950	
	Literal	951	
	Literal	952	
	Literal	953	
	Literal	955	
	Literal	956	
	Literal	958	
	Literal	959	
	Literal	960	
	Literal	961	
	Literal	962	
	Literal	963	
	Literal	964	
	Literal	966	
	Literal	967	
	Literal	968	
	Literal	969	
	Literal	971	
	Literal	972	
	Literal	973	
	Literal	974	
	Literal	975	
	Literal	976	
	Literal	977	
	Literal	981	
DS$K_TYPE_ERROR_END	Literal	84	
	Literal	793	
	Literal	804	
	Literal	829	
	Literal	935	
	Literal	936	
	Literal	937	
	Literal	939	
	Literal	940	
	Literal	941	
	Literal	943	
	Literal	944	
	Literal	945	
	Literal	946	
	Literal	947	
	Literal	948	
	Literal	950	
	Literal	951	
	Literal	952	
	Literal	953	
	Literal	955	
	Literal	956	
	Literal	958	
	Literal	959	
	Literal	960	
	Literal	961	
	Literal	962	
	Literal	963	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_ERRPREP  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------

    Literal     981
DS$K_TYPE_ERRSOFT  Literal      84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_ERRSUP  Literal      84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_ERRSYS  Litera      84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_ERR_HALT  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_EXCEPTION  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_EXCEPTION_HEAD Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_FIRST_PASS  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
```

Symbol	Type	Defined	Referenced ...
Literal			952
Literal			953
Literal			955
Literal			956
Literal			958
Literal			959
Literal			960
Literal			961
Literal			962
Literal			963
Literal			964
Literal			966
Literal			967
Literal			968
Literal			969
Literal			971
Literal			972
Literal			973
Literal			974
Literal			975
Literal			976
Literal			977
Literal			981
DS$K_TYPE_GENERAL	Literal	84	645
Literal			793
Literal			804
Literal			829
Literal			935
Literal			936
Literal			937
Literal			939
Literal			940
Literal			941
Literal			943
Literal			944
Literal			945
Literal			946
Literal			947
Literal			948
Literal			950
Literal			951
Literal			952
Literal			953
Literal			955
Literal			956
Literal			958
Literal			959
Literal			960
Literal			961
Literal			962
Literal			963
Literal			964
Literal			966

Symbol	Type	Defined	Referenced ...
	Literal		967
	Literal		968
	Literal		969
	Literal		971
	Literal		972
	Literal		973
	Literal		974
	Literal		975
	Literal		976
	Literal		977
	Literal		981
DS$K_TYPE_GENERAL_ERROR	Literal		84
	Literal		793
	Literal		804
	Literal		829
	Literal		935
	Literal		936
	Literal		937
	Literal		939
	Literal		940
	Literal		941
	Literal		943
	Literal		944
	Literal		945
	Literal		946
	Literal		947
	Literal		948
	Literal		950
	Literal		951
	Literal		952
	Literal		953
	Literal		955
	Literal		956
	Literal		958
	Literal		959
	Literal		960
	Literal		961
	Literal		962
	Literal		963
	Literal		964
	Literal		966
	Literal		967
	Literal		968
	Literal		969
	Literal		971
	Literal		972
	Literal		973
	Literal		974
	Literal		975
	Literal		976
	Literal		977
	Literal		981
DS$K_TYPE_NO_TESTS	Literal		84

Symbol	Type	Defined	Referenced ...
	Literal	793	
	Literal	804	
	Literal	829	
	Literal	935	
	Literal	936	
	Literal	937	
	Literal	939	
	Literal	940	
	Literal	941	
	Literal	943	
	Literal	944	
	Literal	945	
	Literal	946	
	Literal	947	
	Literal	948	
	Literal	950	
	Literal	951	
	Literal	952	
	Literal	953	
	Literal	955	
	Literal	956	
	Literal	958	
	Literal	959	
	Literal	960	
	Literal	961	
	Literal	962	
	Literal	963	
	Literal	964	
	Literal	966	
	Literal	967	
	Literal	968	
	Literal	969	
	Literal	971	
	Literal	972	
	Literal	973	
	Literal	974	
	Literal	975	
	Literal	976	
	Literal	977	
	Literal	981	
DS$K_TYPE_PARAM_ERROR	Literal	84	
	Literal	793	
	Literal	804	
	Literal	829	
	Literal	935	
	Literal	936	
	Literal	937	
	Literal	939	
	Literal	940	
	Literal	941	
	Literal	943	
	Literal	944	
	Literal	945	

Symbol	Type	Defined	Referenced ...
	Literal	946	
	Literal	947	
	Literal	948	
	Literal	950	
	Literal	951	
	Literal	952	
	Literal	953	
	Literal	955	
	Literal	956	
	Literal	958	
	Literal	959	
	Literal	960	
	Literal	961	
	Literal	962	
	Literal	963	
	Literal	964	
	Literal	966	
	Literal	967	
	Literal	968	
	Literal	969	
	Literal	971	
	Literal	972	
	Literal	973	
	Literal	974	
	Literal	975	
	Literal	976	
	Literal	977	
	Literal	981	
DS$K_TYPE_PROGRAM_END	Literal	84	
	Literal	793	
	Literal	804	
	Literal	829	
	Literal	935	
	Literal	936	
	Literal	937	
	Literal	939	
	Literal	940	
	Literal	941	
	Literal	943	
	Literal	944	
	Literal	945	
	Literal	946	
	Literal	947	
	Literal	948	
	Literal	950	
	Literal	951	
	Literal	952	
	Literal	953	
	Literal	955	
	Literal	956	
	Literal	958	
	Literal	959	
	Literal	960	

Symbol	Type	Defined	Referenced ...
	Literal	961	
	Literal	962	
	Literal	963	
	Literal	964	
	Literal	966	
	Literal	967	
	Literal	968	
	Literal	969	
	Literal	971	
	Literal	972	
	Literal	973	
	Literal	974	
	Literal	975	
	Literal	976	
	Literal	977	
	Literal	981	
DS$K_TYPE_PROGRAM_INFO	Literal	84	
	Literal	793	
	Literal	804	
	Literal	829	
	Literal	935	
	Literal	936	
	Literal	937	
	Literal	939	
	Literal	940	
	Literal	941	
	Literal	943	
	Literal	944	
	Literal	945	
	Literal	946	
	Literal	947	
	Literal	948	
	Literal	950	
	Literal	951	
	Literal	952	
	Literal	953	
	Literal	955	
	Literal	956	
	Literal	958	
	Literal	959	
	Literal	960	
	Literal	961	
	Literal	962	
	Literal	963	
	Literal	964	
	Literal	966	
	Literal	967	
	Literal	968	
	Literal	969	
	Literal	971	
	Literal	972	
	Literal	973	
	Literal	974	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------

    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_PROGRAM_START  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_QIO_INVADP  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
```

Symbol	Type	Defined	Referenced ...
	Literal		940
	Literal		941
	Literal		943
	Literal		944
	Literal		945
	Literal		946
	Literal		947
	Literal		948
	Literal		950
	Literal		951
	Literal		952
	Literal		953
	Literal		955
	Literal		956
	Literal		958
	Literal		959
	Literal		960
	Literal		961
	Literal		962
	Literal		963
	Literal		964
	Literal		966
	Literal		967
	Literal		968
	Literal		969
	Literal		971
	Literal		972
	Literal		973
	Literal		974
	Literal		975
	Literal		976
	Literal		977
	Literal		981
DS$K_TYPE_QIO_NODRIVER	Literal		84
	Literal		793
	Literal		804
	Literal		829
	Literal		935
	Literal		936
	Literal		937
	Literal		939
	Literal		940
	Literal		941
	Literal		943
	Literal		944
	Literal		945
	Literal		946
	Literal		947
	Literal		948
	Literal		950
	Literal		951
	Literal		952
	Literal		953

Symbol	Type	Defined	Referenced ...
	Literal		955
	Literal		956
	Literal		958
	Literal		959
	Literal		960
	Literal		961
	Literal		962
	Literal		963
	Literal		964
	Literal		966
	Literal		967
	Literal		968
	Literal		969
	Literal		971
	Literal		972
	Literal		973
	Literal		974
	Literal		975
	Literal		976
	Literal		977
	Literal		981
DS$K_TYPE_QIO_WRONGVER	Literal		84
	Literal		793
	Literal		804
	Literal		829
	Literal		935
	Literal		936
	Literal		937
	Literal		939
	Literal		940
	Literal		941
	Literal		943
	Literal		944
	Literal		945
	Literal		946
	Literal		947
	Literal		948
	Literal		950
	Literal		951
	Literal		952
	Literal		953
	Literal		955
	Literal		956
	Literal		958
	Literal		959
	Literal		960
	Literal		961
	Literal		962
	Literal		963
	Literal		964
	Literal		966
	Literal		967
	Literal		968

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_SCRIPT_ECHO  Literal      84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_SCRIPT_PNF  Literal      84
    Literal     793
    Literal     804
```

```
Symbol                          Type  Defined  Referenced ...
------ ------------------------ -------- ----- ---------------
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_SCRIPT_PROMPT  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_SCRIPT_SKIP  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_SEQUENCE_ERROR Literal      84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     977
    Literal     981
DS$K_TYPE_START_ERR  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
    Literal     943
    Literal     944
    Literal     945
    Literal     946
    Literal     947
    Literal     948
    Literal     950
    Literal     951
    Literal     952
    Literal     953
    Literal     955
    Literal     956
    Literal     958
    Literal     959
    Literal     960
    Literal     961
    Literal     962
    Literal     963
    Literal     964
    Literal     966
    Literal     967
    Literal     968
    Literal     969
    Literal     971
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$K_TYPE_START_LIST  Literal     84
    Literal     793
    Literal     804
    Literal     829
    Literal     935
    Literal     936
    Literal     937
    Literal     939
    Literal     940
    Literal     941
```

Symbol	Type	Defined	Referenced ...
	Literal	943	
	Literal	944	
	Literal	945	
	Literal	946	
	Literal	947	
	Literal	948	
	Literal	950	
	Literal	951	
	Literal	952	
	Literal	953	
	Literal	955	
	Literal	956	
	Literal	958	
	Literal	959	
	Literal	960	
	Literal	961	
	Literal	962	
	Literal	963	
	Literal	964	
	Literal	966	
	Literal	967	
	Literal	968	
	Literal	969	
	Literal	971	
	Literal	972	
	Literal	973	
	Literal	974	
	Literal	975	
	Literal	976	
	Literal	977	
	Literal	981	
DS$K_TYPE_SUMMARY	Literal	84	
	Literal	793	
	Literal	804	
	Literal	829	
	Literal	935	
	Literal	936	
	Literal	937	
	Literal	939	
	Literal	940	
	Literal	941	
	Literal	943	
	Literal	944	
	Literal	945	
	Literal	946	
	Literal	947	
	Literal	948	
	Literal	950	
	Literal	951	
	Literal	952	
	Literal	953	
	Literal	955	
	Literal	956	

Symbol	Type	Defined	Referenced ...
	Literal		958
	Literal		959
	Literal		960
	Literal		961
	Literal		962
	Literal		963
	Literal		964
	Literal		966
	Literal		967
	Literal		968
	Literal		969
	Literal		971
	Literal		972
	Literal		973
	Literal		974
	Literal		975
	Literal		976
	Literal		977
	Literal		981
DS$K_TYPE_USER_PROMPT	Literal		84
	Literal		793
	Literal		804
	Literal		829
	Literal		935
	Literal		936
	Literal		937
	Literal		939
	Literal		940
	Literal		941
	Literal		943
	Literal		944
	Literal		945
	Literal		946
	Literal		947
	Literal		948
	Literal		950
	Literal		951
	Literal		952
	Literal		953
	Literal		955
	Literal		956
	Literal		958
	Literal		959
	Literal		960
	Literal		961
	Literal		962
	Literal		963
	Literal		964
	Literal		966
	Literal		967
	Literal		968
	Literal		969
	Literal		971

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     972
    Literal     973
    Literal     974
    Literal     975
    Literal     976
    Literal     977
    Literal     981
DS$LOAD     GlobLit      88
DS$LOADPCS    GlobLit      88
DS$MAPDBGBLOCK    GlobLit      88
DS$MEMSIZE    GlobLit      88
DS$MMOFF    GlobLit      88
DS$MMON     GlobLit      88
DS$PARSE    GlobLit      88
DS$PRINTB    GlobLit      88
DS$PRINTF    GlobLit      88
DS$PRINTS    GlobLit      88
DS$PRINTSIG    GlobLit      88
DS$PRINTX    GlobLit      88
DS$PROBE    GlobLit      88
DS$RELBUF    GlobLit      88
DS$SERVICE_DISPATCHER  GlobLit      88
DS$SETIPL    GlobLit      88
DS$SETMAP    GlobLit      88
DS$SETPRIEXV    GlobLit      88
DS$SETVEC    GlobLit      88
DS$SHOCHAN    GlobLit      88
DS$SUMMARY    GlobLit      88
DS$WAITMS    GlobLit      88
DS$WAITUS    GlobLit      88
FILL_FULL_ROW    Macro       165   284     479
FILL_N_ENTRYS    Unbound             156     170     190     205     222     257     266     281     293     302
                311     320     328     336     347     362     371     380     385     394
                399     408     417     427     442     457     462     467     476     488
                497     515     530     545     560     575     590     605     614
    Macro        136   222     257     266     281     284     293     302     311     320     328
                336     347     362     371     380     385     394     399     408     417
                427     442     457     462     467     476     479     488     497     500
                515     530     545     560     575     590     605     614
GET1ST_     Macro    Lib04     84      85     793     804     829     935     936     937     939     940
                941     943     944     945     946     947     948     950     951     952
                953     955     956     958     959     960     961     962     963     964
                966     967     968     969     971     972     973     974     975     976
                977     981
GET2ND_     Unbound             84      85     793     804     829     935     936     937     939     940
                941     943     944     945     946     947     948     950     951     952
                953     955     956     958     959     960     961     962     963     964
                966     967     968     969     971     972     973     974     975     976
                977     981
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal      85
HS$K_ACTION_ADVANCE_GENERAL_COM Literal      85
HS$K_ACTION_ADVANCE_STATE Literal      85
HS$K_ACTION_CHANGE_TABLE Literal      85
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_ACTION_DIAGNOSTIC_ERROR Literal     85
HS$K_ACTION_DONE_GENERAL_COMS Literal     85
HS$K_ACTION_DO_NOTHING  Literal     85
HS$K_ACTION_DUMMY_3  Literal     85
HS$K_ACTION_DUMMY_4  Literal     85
HS$K_ACTION_DUMMY_5  Literal     85
HS$K_ACTION_DUMMY_6  Literal     85
HS$K_ACTION_INIT_HS  Literal     85
HS$K_ACTION_INTERNAL_ERROR Literal     85
HS$K_ACTION_LAST_ACTION  Literal     85
HS$K_ACTION_LOAD_ERROR  Literal     85
HS$K_ACTION_LOAD_GENERAL_COM Literal     85
HS$K_ACTION_LOAD_LOAD_COM Literal     85
HS$K_ACTION_LOAD_SELECT_COM Literal     85
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     85
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     85
HS$K_ACTION_LOAD_START_COM Literal     85
HS$K_ACTION_PREPARATION_ERROR Literal     85
HS$K_ACTION_START_ERROR  Literal     85
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     85
HS$K_STATE_ADVANCE_GENERAL_COM Literal     85
HS$K_STATE_CHANGE_TABLE  Literal     85
HS$K_STATE_DIAGNOSTIC_ERROR Literal     85
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     85
HS$K_STATE_DONE_GENERAL_COMS Literal     85
HS$K_STATE_DUMMY_1  Literal     85
HS$K_STATE_DUMMY_2  Literal     85
HS$K_STATE_DUMMY_3  Literal     85
HS$K_STATE_DUMMY_4  Literal     85
HS$K_STATE_DUMMY_6  Literal     85
HS$K_STATE_INIT_HS  Literal     85
HS$K_STATE_INTERNAL_ERROR Literal     85
HS$K_STATE_LAST_STATE  Literal     85
HS$K_STATE_LOAD_ERROR  Literal     85
HS$K_STATE_LOAD_GENERAL_COM Literal     85
HS$K_STATE_LOAD_LOAD_COM Literal     85
HS$K_STATE_LOAD_SELECT_COM Literal     85
HS$K_STATE_LOAD_SET_EVENT_COM Literal     85
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     85
HS$K_STATE_LOAD_START_COM Literal     85
HS$K_STATE_PREPARATION_ERROR Literal     85
HS$K_STATE_START_ERROR  Literal     85
INSERT_ENTRYS   Unbound          160     193     222     257     266     281     293     302     311     320
              328     336     347     362     371     380     385     394     399     408
              417     427     442     457     462     467     476     488     497     500
              515     530     545     560     575     590     605     614
    Macro      155   222     257     266     281     293     302     311     320     328     336
              347     362     371     380     385     394     399     408     417     427
              442     457     462     467     476     488     497     500     515     530
              545     560     575     590     605     614
INSERT_N    Unbound          199     222     257     266     281     293     302     311     320     328
              336     347     362     371     380     385     394     399     408     417
              427     442     457     462     467     476     488     497     500     515
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
                 530     545     560     575     590     605     614
    Macro        180   216     222     225     257     260     266     269     281     287     293
                 296     302     305     311     314     320     323     328     331     336
                 339     347     350     362     365     371     374     380     383     385
                 388     394     397     399     402     408     411     417     420     427
                 430     442     445     457     460     462     465     467     470     476
                 482     488     491     497     500     503     515     518     530     533
                 545     548     560     563     575     578     590     593     605     608
                 614
INTERNAL_ERROR   Unbound          170     190     205     222     257     266     281     293     302     311
                 320     328     336     347     362     371     380     385     394     399
                 408     417     427     442     457     462     467     476     488     497
                 515     530     545     560     575     590     605     614
LENGTH    Unbound          778
    RoutForm     713   804.    844.    846.    847.    848.    849.    851.    852.    853.    855.
                 857.    858.    859.    860.    861.    863.    864.    866.    867.    868.
                 869.    870.    871.    888.    894.
MEN$K_HS_STATE_TABLE  Literal     85
MEN$K_MM_STATE_TABLE  Literal     85
MEN$K_TQ_STATE_TABLE  Literal     85
MENU$K_EXIT_AUTOSIZER_ERROR Literal     85
MENU$K_EXIT_INTERNAL_ERROR Literal     85
MENU$K_EXIT_LAST_REASON  Literal     85
MENU$K_EXIT_OPERATOR_ABORT Literal     85
MENU$K_EXIT_OPERATOR_STOP Literal     85
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     85
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     85
MM$K_ACTION_ADVANCE_STATE Literal     85
MM$K_ACTION_AUTOSIZER_ERROR Literal     85
MM$K_ACTION_BUILD_DDBS  Literal     85
MM$K_ACTION_BUILD_HSTS  Literal     85
MM$K_ACTION_CHANGE_TABLE Literal     85
MM$K_ACTION_DONE_INITS  Literal     85
MM$K_ACTION_DO_NOTHING  Literal     85
MM$K_ACTION_DUMMY_3  Literal     85
MM$K_ACTION_DUMMY_4  Literal     85
MM$K_ACTION_DUMMY_5  Literal     85
MM$K_ACTION_DUMMY_6  Literal     85
MM$K_ACTION_DUMMY_7  Literal     85
MM$K_ACTION_DUMMY_8  Literal     85
MM$K_ACTION_INTERNAL_ERROR Literal     85
MM$K_ACTION_LAST_ACTION  Literal     85
MM$K_ACTION_LOAD_AUTOSIZER Literal     85
MM$K_ACTION_MENU_PROMPT  Literal     85
MM$K_ACTION_PRINT_MENU  Literal     85
MM$K_ACTION_PRINT_MENU_HEADING Literal     85
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     85
MM$K_ACTION_START_AUTOSIZER Literal     85
MM$K_STATE_AUTOSIZER_ERROR Literal     85
MM$K_STATE_AUTOSIZER_RUNNING Literal     85
MM$K_STATE_BUILD_DDBS  Literal     85
MM$K_STATE_BUILD_HSTS  Literal     85
MM$K_STATE_CHANGE_TABLE  Literal     85
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_STATE_DONE_INITS  Literal     85
MM$K_STATE_DUMMY_1  Literal     85
MM$K_STATE_DUMMY_2  Literal     85
MM$K_STATE_DUMMY_3  Literal     85
MM$K_STATE_DUMMY_5  Literal     85
MM$K_STATE_DUMMY_7  Literal     85
MM$K_STATE_DUMMY_8  Literal     85
MM$K_STATE_INTERNAL_ERROR Literal     85
MM$K_STATE_LAST_STATE  Literal     85
MM$K_STATE_LOAD_AUTOSIZER Literal     85
MM$K_STATE_MENU_PROMPT  Literal     85
MM$K_STATE_PRINT_MENU  Literal     85
MM$K_STATE_PRINT_MENU_HEADING Literal     85
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     85
MM$K_STATE_START  Literal     85
MM$K_STATE_START_AUTOSIZER Literal     85
N     MacrForm   136   137     140     145     149
NEW_STATE   MacrForm   136   138
    MacrForm   155   156
NUL2ND_     Macro    Lib04    84      85     793     804     829     935     936     937     939     940
              941     943     944     945     946     947     948     950     951     952
              953     955     956     958     959     960     961     962     963     964
              966     967     968     969     971     972     973     974     975     976
              977     981
RECURSION   Unbound          779     784
    Own        752   752=    793.    795=    795.    833=    833.    844=    844.    846=    846.
              847=    847.    848=    848.    849=    849.    851=    851.    852=    852.
              853=    853.    855=    855.    857=    857.    858=    858.    859=    859.
              860=    860.    861=    861.    863=    863.    864=    864.    866=    866.
              867=    867.    868=    868.    869=    869.    870=    870.    871=    871.
STATE    MacrForm   180   193
    Register   812   820=    823.
STATE_INTERNAL_ERROR  Unbound          170     190     205     222     257     266     281     293     302     311
              320     328     336     347     362     371     380     385     394     399
              408     417     427     442     457     462     467     476     488     497
              515     530     545     560     575     590     605     614
STATE_TABLE_STRUCTURE  Structur Lib03   617
STATUS    Unbound          759     774     776     778     780     785
    Local      844   844=    844.
    Local      846   846=    846.
    Local      847   847=    847.
    Local      848   848=    848.
    Local      849   849=    849.
    Local      851   851=    851.
    Local      852   852=    852.
    Local      853   853=    853.
    Local      855   855=    855.
    Local      857   857=    857.
    Local      858   858=    858.
    Local      859   859=    859.
    Local      860   860=    860.
    Local      861   861=    861.
    Local      863   863=    863.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Local       864   864=    864.
    Local       866   866=    866.
    Local       867   867=    867.
    Local       868   868=    868.
    Local       869   869=    869.
    Local       870   870=    870.
    Local       871   871=    871.
SYS$ALLOC   GlobLit     88
SYS$ASCTIM   GlobLit     88
SYS$ASSIGN   GlobLit     88
SYS$BINTIM   GlobLit     88
SYS$CANCEL   GlobLit     88
SYS$CANTIM   GlobLit     88
SYS$CLOSE   GlobLit     88
SYS$CLREF   GlobLit     88
SYS$CONNECT   GlobLit     88
SYS$DALLOC   GlobLit     88
SYS$DASSGN   GlobLit     88
SYS$DISCONNECT   GlobLit     88
SYS$FAO    GlobLit     88
SYS$FAOL   GlobLit     88
SYS$GET    GlobLit     88
SYS$GETCHN   GlobLit     88
SYS$GETTIM   GlobLit     88
SYS$GTCHAN   GlobLit     88
SYS$HIBER   GlobLit     88
SYS$LKWSET   GlobLit     88
SYS$NUMTIM   GlobLit     88
SYS$OPEN   GlobLit     88
SYS$QIO    GlobLit     88
SYS$QIOW   GlobLit     88
SYS$READ   GlobLit     88
SYS$READEF   GlobLit     88
SYS$SETAST   GlobLit     88
SYS$SETEF   GlobLit     88
SYS$SETIMR   GlobLit     88
SYS$SETPRI   GlobLit     88
SYS$SETPRT   GlobLit     88
SYS$SETRWN   GlobLit     88
SYS$ULKPAG   GlobLit     88
SYS$ULWSET   GlobLit     88
SYS$UNWIND   GlobLit     88
SYS$WAITFR   GlobLit     88
SYS$WFLAND   GlobLit     88
SYS$WFLOR   GlobLit     88
TQ$K_ACTION_ADVANCE_STATE Literal     85
TQ$K_ACTION_CHANGE_TABLE Literal     85
TQ$K_ACTION_DONE_TEST_QUEUE Literal     85
TQ$K_ACTION_DO_NOTHING  Literal     85
TQ$K_ACTION_DUMMY_3  Literal     85
TQ$K_ACTION_DUMMY_4  Literal     85
TQ$K_ACTION_DUMMY_5  Literal     85
TQ$K_ACTION_GET_DDB  Literal     85
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
TQ$K_ACTION_INIT_TQ  Literal     85
TQ$K_ACTION_INTERNAL_ERROR Literal     85
TQ$K_ACTION_LAST_ACTION  Literal     85
TQ$K_STATE_CHANGE_TABLE  Literal     85
TQ$K_STATE_DONE_TEST_QUEUE Literal     85
TQ$K_STATE_DUMMY_1  Literal     85
TQ$K_STATE_DUMMY_2  Literal     85
TQ$K_STATE_DUMMY_3  Literal     85
TQ$K_STATE_DUMMY_4  Literal     85
TQ$K_STATE_DUMMY_5  Literal     85
TQ$K_STATE_GET_DDB  Literal     85
TQ$K_STATE_INIT_TQ  Literal     85
TQ$K_STATE_INTERNAL_ERROR Literal     85
TQ$K_STATE_LAST_STATE  Literal     85
TYPECODE   Unbound          778
    RoutForm   713   799.   804.   807.   844.   846.   847.   848.   849.   851.   852.
               853.   855.   857.   858.   859.   860.   861.   863.   864.   866.
               867.   868.   869.   870.   871.



  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDSTATE.B32;1
    2  Module CRDSTATE
   72  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   75  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   78  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   81  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  985  Eludom CRDSTATE


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     @ Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
```

```
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;            671          0          0         46           00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;            923          4          0         94           00:00.1
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;            486         45          9         62           00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0      1000           00:04.4




;     COMMAND QUALIFIERS

; BLISS CRDSTATE.B32/LIST=CRDSTATE.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDSTATE.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  1250 code + 6610 data bytes
; Run Time:    02:57.2
; Elapsed Time:    06:32.6
; Lines/CPU Min:     333
; Lexemes/CPU-Min: 83671
; Memory Used: 877 pages
; Compilation Complete
```

```
;    0001   0 %TITLE '*** CRDSTOP Module'
;    0002   0 MODULE CRDSTOP (    ! Routines to handle stopping CRD
;    0003   0   IDENT = 'V01.20'  ) =
;    0004   0 !++
;    0005   0 ! COPYRIGHT (c) 1980, 1981
;    0006   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0007   0 !
;    0008   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0009   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0010   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0011   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0012   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0013   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0014   0 ! REMAIN IN DEC.
;    0015   0 !
;    0016   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0017   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0018   0 ! CORPORATION.
;    0019   0 !
;    0020   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0021   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0022   0 !-
;    0023   0 !++
;    0024   0 ! Facility:
;    0025   0 !
;    0026   0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0027   0 !
;    0028   0 ! Abstract:
;    0029   0 !
;    0030   0 ! This module contains those routine(s) necessary to stop the Customer Runnable
;    0031   0 ! Diagnostics image.
;    0032   0 !
;    0033   0 ! Author:
;    0034   0 !
;    0035   0 ! Jack Stansbury
;    0036   0 !
;    0037   0 ! Creation Date:
;    0038   0 !
;    0039   0 ! April 13, 1982
;    0040   0 !
;    0041   0 ! Version:
;    0042   0 !
;    0043   0 ! 6.9
;    0044   0 !
;    0045   0 ! Modified By:
;    0046   0 !
;    0047   0 ! 01 Jack Stansbury, 2-July-1982, Version 1.0
;    0048   0 !  Added code in CRD$Stop routine at the end to transfer what is in the CRD$GL_CRD$Debug_Vector
;    0049   0 !  to the dispatch vector area. This transfer must take place AFTER all $CRD_Print's have taken
;    0050   0 !  place!!!!!
;    0051   0 !
;    0052   0 ! 02 Jack Stansbury, Version 1.0, July 9, 1982
;    0053   0 !  Added the AutoTest_End_Message to the Internal_Error stop message.
;    0054   0 !
;    0055   0 ! 03 Jack Stansbury, Version 1.1, July 25, 1982
```

```
;    0056  0 !  Changed the printing of the messages to be compatible with Menu Test. This entails adding the
;    0057  0 !  CRD$GT_Menu and CRD$GT_Auto messages to the CRDMESSAG module, and using these when this routine
;    0058  0 !  prints messages.
;    0059  0 !
;    0060  0 ! 04 Jack Stansbury, Version 1.1, July 27, 1982
;    0061  0 !  Added code to dispose of the Device Data Blocks before CRD$Stop exits. This is so that the memory
;    0062  0 !
;    0063  0 ! 05 Jack Stansbury, Version 1.1, July 27, 1982
;    0064  0 !  Added two mode CRD$K_Exit_CRD reasons for stopping CRD. These are for Menu Test.
;    0065  0 !
;    0066  0 ! 06 Jack Stansbury, Version 1.1, August 20, 1982
;    0067  0 !  Changed the fields in the DDBs.
;    0068  0 !
;    0069  0 ! 07 Jack Stansbury, Version 1.1, August 20, 1982
;    0070  0 !  Added code to call the Menu_Cleanup routine - to clean up Menu Test.
;    0071  0 !
;    0072  0 ! 08  John Ciukaj, Version 1.1, Sept 1,1982
;    0073  0 !  Refined 'stop reason' printouts for CRD MENU
;    0074  0 !
;    0075  0 ! 09 Jack Stansbury, Version 1.1, September 12, 1982
;    0076  0 !  Added code to print the correct 'RETURNING TO . . .' message.
;    0077  0 !
;    0078  0 ! 10 John Ciukaj, Version 1.1, September 16,1982
;    0079  0 !   Added call to MEN$FncTst_CmplErr_Status' in
;    0080  0 !  CRD$K_Exit_CRD_Operator stop reason.
;    0081  0 !
;    0082  0 ! 11 Jack Stansbury, Version 1.1, September 21, 1982
;    0083  0 !  Changed the CRD$K_Exit literals to AUTO$K_Exit_ and MENU$K_Exit.
;    0084  0 !  Also changed the spelling of the DDB$V_Status bits.
;    0085  0 !
;    0086  0 ! 12 Jack Stansbury, Version 1.1, September 24, 1982
;    0087  0 !  Changed the code for the above change in mnemonics (11).
;    0088  0 !
;    0089  0 ! 13 Jack Stansbury, Version 1.1, September ??, 1982
;    0090  0 !  ???
;    0091  0 !
;    0092  0 ! 14 Jack Stansbury, Version 1.1, September 28, 1982
;    0093  0 !  Made changes for Auto going to Menu if no devices are bad.
;    0094  0 !
;    0095  0 ! 15 Jack Stansbury, Version ?.?, March 3, 1983
;    0096  0 !  Straigtened out the Debug_Vector stuff - correct length
;    0097  0 !  for the Dispatch_Vectors, etc. Added CRD_Assert statements
;    0098  0 !  for the various dispatch vector lengths.
;    0099  0 !
;    0100  0 ! 16  John Ciukaj Version 1.2 4-Apr-1983
;    0101  0 !  - Changed references to CRD bits in DSA Flags to
;    0102  0 !    distinguish between Online and Offline.
;    0103  0 !  - Added Stop reason for CRD AUTO in CRD$Stop
;    0104  0 !  - Added flag to indicate CRD MENU invoked from completion
;    0105  0 !    of CRD AUTO
;    0106  0 !
;    0107  0 ! 17  Scott Cohen  Version 1.2 9-Jun-1983
;    0108  0 !  For soft console support
;    0109  0 !  - Added new routine CRD$Screen_Pause,
;    0110  0 !  -  $SCREEN_Literals definitions,
```

```
;    0111  0 !  -  new routine CRD$Auto_Summary,
;    0112  0 !
;    0113  0 ! 18  Scott Cohen  Version 1.3 23-Sep-1983
;    0114  0 !  Added new type of soft console the PEARL
;    0115  0 !
;    0116  0 ! 19 Ron Parker October 22,1984
;    0117  0 !  - Added Menu_test_on handling
;    0118  0 !
;    0119  0 ! 20 Amy Iorio July 16, 1985
;    0120  0 !  - Put screen clear into selectone format and
;    0121  0 !  added handling of VT200.
;    0122  0 !
;    0123  0 !--
;    0124  0 %SBTTL 'Libraries'
;    0125  1 BEGIN     ! Begin the CRDSTOP module
;    0126  1
;    0127  1 LIBRARY
;    0128  1  '$DS';    ! Use DS.L32 first
;    0129  1
;    0130  1 LIBRARY
;    0131  1  '$Diag';   ! Use Diag.L32 second
;    0132  1
;    0133  1 LIBRARY
;    0134  1  '$CRD';    ! Use CRD.L32 third
;    0135  1
;    0136  1 LIBRARY
;    0137  1  'Sys$Library:Lib';  ! Use Lib as last resort
;    0138  1 %SBTTL 'Table of Contents'
;    0139  1
;    0140  1 FORWARD ROUTINE
;    0141  1  CRD$Control_C_Handler  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0142  1      ! Handles Control-C's during the running of CRD
;    0143  1
;    0144  1  CRD$Stop   : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0145  1      ! Stops the CRD process.
;    0146  1
;    0147  1  CRD$Auto_Summary  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0148  1      ! Print a summary of Auto Test.
;    0149  1
;    0150  1  CRD$Screen_Pause   : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
;    0151  1      ! Handles soft console issues.
;    0152  1 %SBTTL 'Included Macros and Literals'
;    0153  1
;    0154  1  $CRD_Literals;   ! Define the CRD literals
;    0155  1  $DS_DSADef;   ! Define the DSA locations
;    0156  1  $SCREEN_Literals;  ! Define the SCREEN operation literals
;    0157  1 %SBTTL 'Psect Definitions'
;    0158  1
;    0159  1 PSECT
;    0160  1     PLIT   = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0161  1
;    0162  1 PSECT
;    0163  1     CODE   = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG RELATIVE), PIC);
;    0164  1
;    0165  1 PSECT
```

```
;    0166  1      OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0167  1
;    0168  1 PSECT
;    0169  1      GLOBAL = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0170  1 %SBTTL 'Module-Global Static Storage'
;    0171  1
;    0172  1  ModNam ('CRDSTOP');  ! Module name for ErrSup macro
;    0173  1 %SBTTL 'Module-Global Static Storage'
;    0174  1
;    0175  1 GLOBAL
;    0176  1  CRD$GB_Auto_To_Menu_Off : BYTE INITIAL (BYTE(False)),
;    0177  1      ! Flag indicates that CRD MENU Offline
;    0178  1      ! was invoked from completion of CRD AUTO Offline     [16]
;    0179  1  CRD$GL_CRD_Debug_Vector : VECTOR [CRD$K_Total_Numb_Vectors, LONG];     ![15]
;    0180  1       ! Storage for debugging information that will be printed by the
;    0181  1       ! DSV$Exit routine (in the Resident-DS).
```

```
;    0182  1 %SBTTL 'CRD$Stop Routine'
;    0183  1
;    0184  1 GLOBAL ROUTINE CRD$Stop (Stop_Reason) : NOVALUE =
;    0185  1
;    0186  1 !++
;    0187  1 ! Functional Description:
;    0188  1 !
;    0189  1 ! Stop the CRD image in its tracks! Exit to either Console mode of to the Supervisor.   ![03]
;    0190  1 !
;    0191  1 ! Formal Input Parameters:
;    0192  1 !
;    0193  1 ! Stop_Reason : The reason for stopping CRD.
;    0194  1 !
;    0195  1 ! Formal Output Parameters:
;    0196  1 !
;    0197  1 ! None
;    0198  1 !
;    0199  1 ! Side Effects:
;    0200  1 !
;    0201  1 ! CRD (and the Resident-DS) is stopped.
;    0202  1 !
;    0203  1 ! Completion Codes:
;    0204  1 !
;    0205  1 ! None
;    0206  1 !--
;    0207  2 BEGIN     ! Routine CRD$Stop
;    0208  2  BIND
;    0209  2   CRD_Dispatch_Vectors = (CRD$AL_CRD_Dispatch_Vectors + 4) : VECTOR [, LONG];
;    0210  2      ! Reference the vectors as an array of longwords
;    0211  2
;    0212  2  ROUTINE Print_Exit_To : NOVALUE =         ![09]
;    0213  3  BEGIN              ![09]
;    0214  3  BIND ROUTINE             ![09]
;    0215  3   DSR$Check_MenuTest_Off_Set = (.CRD$DSR$Check_MenuTest_Off_Set) : JSB_None,!   [16]
;    0216  3   DSR$Check_AutoTest_Off_Set = (.CRD$DSR$Check_AutoTest_Off_Set) : JSB_None; !   [16]
;    0217  3
;    0218  4  IF ( DSR$Check_AutoTest_Off_Set () OR
;    0219  4       ! CRD AUTO
;    0220  4   (DSR$Check_MenuTest_Off_Set () EQL 3) OR
;    0221  4       ! CRD MENU invoked by 'T/M'
;    0222  4       ! console command
;    0223  3   .CRD$GB_Auto_To_Menu_Off) THEN ! CRD MENU invoked from CRD AUTO     [16]
;    0224  3
;    0225  4   $CRD_Print (CRD$GT_Exit_To_Console)        ![09]
;    0226  3  ELSE              ![09]
;    0227  3   $CRD_Print (CRD$GT_Exit_To_CLI);        ![09]
;    0228  3   CRD$GB_Auto_To_Menu_Off = False; ! Init Flag        [16]
;    0229  2  END;              ![09]


        .TITLE  CRDSTOP *** CRDSTOP Module
        .IDENT  \V01.20\

        .PSECT  WORK,NOEXE,2
```

```
        00  00000 CRD$GB_AUTO_TO_MENU_OFF::
            .BYTE   0               ;
            00001   .BLKB   3
            00004 CRD$GL_CRD_DEBUG_VECTOR::
            .BLKB   100

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

        50  4F  54  53  44  52  43  07  00000 P.AAA:  .ASCII  <7>\CRDSTOP\           ;

           DSA$AL_APTMAIL=     65024
           DSA$GL_FLAGS=       65024
           DSA$GL_APTCOM=      65028
           DSA$GL_PASSES=      65032
           DSA$GL_UNITS=       65036
           DSA$GL_SECTNO=      65040
           DSA$GL_SID=        65044
           DSA$GL_MSGTYP=      65088
           DSA$GL_ERRNO=       65092
           DSA$GL_EVENT=       65096
           DSA$GL_SUBTNO=      65100
           DSA$GL_TESTNO=      65104
           DSA$GL_PASSNO=      65108
           DSA$GL_DEVLEN=      65112
           DSA$GL_DEVNAM=      65116
           DSA$GL_MSGPTR=      65128
           $MODULE=        P.AAA
            .EXTRN  CRD$CONTROL_C_HANDLER
            .EXTRN  CRD$STOP, CRD$AUTO_SUMMARY
            .EXTRN  CRD$SCREEN_PAUSE
            .EXTRN  CRD$AL_CRD_DISPATCH_VECTORS
            .EXTRN  CRD$DSR$CHECK_MENUTEST_OFF_SET
            .EXTRN  CRD$DSR$CHECK_AUTOTEST_OFF_SET
            .EXTRN  CRD$GT_EXIT_TO_CONSOLE
            .EXTRN  CRD$PRINT, CRD$GT_EXIT_TO_CLI

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              OFFC 00000 PRINT_EXIT_TO:
            .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 0212
            52 00000000G   EF   D0 00002    MOVL    CRD$DSR$CHECK_MENUTEST_OFF_SET, R2       ; 0215
00000000G   FF   16 00009    JSB     @CRD$DSR$CHECK_AUTOTEST_OFF_SET        ; 0218
            0E      50   E8 0000F    BLBS    R0, 1$          ;
            62   16 00012    JSB     (R2)            ; 0220
            03      50   D1 00014    CMPL    R0, #3          ;
            07   13 00017    BEQL    1$              ;
            08 00000000'   EF   E9 00019    BLBC    CRD$GB_AUTO_TO_MENU_OFF, 2$         ; 0223
00000000G   EF   9F 00020 1$:   PUSHAB  CRD$GT_EXIT_TO_CONSOLE          ; 0225
            06   11 00026    BRB     3$              ;
00000000G   EF   9F 00028 2$:   PUSHAB  CRD$GT_EXIT_TO_CLI          ; 0227
            01   DD 0002E 3$:   PUSHL   #1             ;
            1A   DD 00030    PUSHL   #26             ;
00000000G   FF      03   FB 00032    CALLS   #3, @CRD$PRINT          ;
00000000'   EF   94 00039    CLRB    CRD$GB_AUTO_TO_MENU_OFF         ; 0228
             04 0003F    RET                 ; 0229
```

```
; Routine Size:  64 bytes,    Routine Base:  CODE + 0000


;    0230  2  CRD$Disable_Ctrl_C ();              ![12]
;    0231  2  CRD$Clear_Ctrl_C_Flag ();            ![12]
;    0232  2
;    0233  2
;    0234  2  SELECTONE .Stop_Reason OF
;    0235  2   SET
;    0236  2   [MENU$K_Exit_Sup_File_Not_Found]:          ![12]
;    0237  3    BEGIN              ![12]
;    0238  3    $CRD_Print (MEN$GT_Support_File_Not_Found, MEN$GT_MenFuncSup_File);   ![12]
;    0239  2    END;               ![12]
;    0240  2
;    0241  2   [MENU$K_Exit_Sup_File_Load_Err]:          ![12]
;    0242  3    BEGIN              ![12]
;    0243  3    $CRD_Print (MEN$GT_Support_File_Load_Err, MEN$GT_MenFuncSup_File);   ![12]
;    0244  2    END;               ![12]
;    0245  2
;    0246  2   [AUTO$K_Exit_Control_C]:           ![11]
;    0247  3    BEGIN
;    0248  3    $CRD_Print (CRD$GT_Control_C_Abort, .CRD$GT_Which_CRD);     ![03]
;    0249  2    END;
;    0250  2
;    0251  2   [MENU$K_Exit_Internal_Error,          ![11]
;    0252  2    AUTO$K_Exit_Internal_Error]:          ![11]
;    0253  3    BEGIN
;    0254  3    $CRD_Print (CRD$GT_Internal_Error, .CRD$GT_Which_CRD);     ![03]
;    0255  3    $CRD_Print (CRD$GT_Internal_Error_Abort, .CRD$GT_Which_CRD);    ![03]
;    0256  2    END;
;    0257  2
;    0258  2   [AUTO$K_Exit_NoErrors_Complete]:         ![11]
;    0259  3    BEGIN
;    0260  3    $CRD_Print (CRD$GT_CRD_No_Errors, .CRD$GT_Which_CRD);     ![03]
;    0261  2    END;
;    0262  2
;    0263  2   [AUTO$K_Exit_Errors_Comp'^^ ]:          ![12]
;    0264  3    BEGIN              ![12]
;    0265  3    $CRD_Print (CRD$GT_CRD_Errors_Complete, .CRD$GT_Which_CRD);    ![12]
;    0266  2    END;
;    0267  2
;    0268  2   [MENU$K_Exit_Operator_Stop]:          ![11]
;    0269  3    BEGIN
;    0270  3    $CRD_Print (MEN$GT_Operator_Stop);        ![12]
;    0271  2    END;
;    0272  2
;    0273  2   [MENU$K_Exit_Operator_Abort]:          ![11]
;    0274  3    BEGIN              ![11]
;    0275  3 !
;    0276  3 ! Take these out for now - add them back later sometime - when I clear up the Complete bit in the
;    0277  3 ! DDB status - problem is that a ^C during testing, and exit to VDS produces MENU TEST COMPLETE
;    0278  3 ! rather than INCOMPLETE.
;    0279  3 !   IF (.MEN$GB_Test_In_Progress)         ![11]
;    0280  3 !   THEN               ![11]
```

```
;    0281  3 !       MEN$FncTst_CmplErr_Status ();       ![11]
;    0282  3 !-
;    0283  3
;    0284  3     $CRD_Print (MEN$GT_Operator_Abort);       ![12]
;    0285  2     END;            ![11]
;    0286  2
;    0287  2   [AUTO$K_Exit_NoErrors_Incomplete]:        ![11]
;    0288  3     BEGIN
;    0289  3     $CRD_Print (CRD$GT_CRD_Incomplete, .CRD$GT_Which_CRD);     ![03]
;    0290  2     END;
;    0291  2
;    0292  2   [AUTO$K_Exit_NoErrors_Prep]:          ![11]
;    0293  3     BEGIN
;    0294  3     $CRD_Print (CRD$GT_CRD_Dev_UnTested, .CRD$GT_Which_CRD);    ![03]
;    0295  2     END;
;    0296  2
;    0297  2   [AUTO$K_Exit_Errors_Incomplete]:        ![11]
;    0298  3     BEGIN
;    0299  3     MAP
;    0300  3       CRD$GA_First_Diagnostic    : REF Device_Data_Block_Structure;
;    0301  3
;    0302  3     LOCAL
;    0303  3       Char_Ptr,
;    0304  3       Numb_Bad_Devices;
;    0305  3
;    0306  3     REGISTER
;    0307  3       DDB_Ptr : REF Device_Data_Block_Structure;
;    0308  3
;    0309  3     !+
;    0310  3     ! Accumulate a list of bad device names in the message buffer.
;    0311  3     !-
;    0312  3
;    0313  3     Char_Ptr   = CH$PTR (CRD$GA_CRD_Message_Buffer + 1);
;    0314  3     Numb_Bad_Devices = 0;
;    0315  3     DDB_Ptr    = .CRD$GA_First_Diagnostic;
;    0316  3
;    0317  3     DO
;    0318  4       BEGIN
;    0319  4       IF (.DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Device_Bad>) THEN   ![11]
;    0320  5         BEGIN
;    0321  5         BIND
;    0322  6           Diag_Vector_Ptr = (.DDB_Ptr [CRD$K_DDB_Auto_Support_Entry])
;    0323  5             : Hardware_Support_Vector;
;    0324  5
;    0325  5         BIND
;    0326  6           Device_Name = (.Diag_Vector_Ptr [CRD$K_Device_Name])
;    0327  5             : VECTOR [, BYTE];
;    0328  5
;    0329  5         Numb_Bad_Devices = .Numb_Bad_Devices + 1;
;    0330  5
;    0331  5         !+
;    0332  5         ! If necessary, move in a   AND '.
;    0333  5         !-
;    0334  5
;    0335  5         IF (.Numb_Bad_Devices GTR 1) THEN
```

```
;    0336  6        BEGIN
;    0337  6        MAP
;    0338  6         CRD$GT_Space_AND_Space : VECTOR [, BYTE];
;    0339  6
;    0340  6        CH$MOVE (
;    0341  6         .CRD$GT_Space_AND_Space [0],
;    0342  6         CH$PTR (CRD$GT_Space_AND_Space [1]),
;    0343  6         .Char_Ptr
;    0344  6         );
;    0345  6        Char_Ptr = CH$PLUS (.Char_Ptr, .CRD$GT_Space_AND_Space [0]);
;    0346  5        END;
;    0347  5
;    0348  5       !+
;    0349  5       ! Move in the actual device name.
;    0350  5       !-
;    0351  5
;    0352  5       CH$MOVE (
;    0353  5         .Device_Name [0],
;    0354  5         CH$PTR (Device_Name [1]),
;    0355  5         .Char_Ptr
;    0356  5         );
;    0357  5       Char_Ptr = CH$PLUS (.Char_Ptr, .Device_Name [0]);
;    0358  4       END;
;    0359  4
;    0360  4      DDB_Ptr = .DDB_Ptr [CRD$K_DDB_FLink];
;    0361  4       ! Advance to the next device.
;    0362  4      END
;    0363  3     UNTIL (.DDB_Ptr EQL .CRD$GA_First_Diagnostic);
;    0364  3
;    0365  3     CRD$GA_CRD_Message_Buffer [0]
;    0366  3      = CH$DIFF (.Char_Ptr, CH$Ptr (CRD$GA_CRD_Message_Buffer + 1));
;    0367  3
;    0368  3     $CRD_Print (CRD$GT_CRD_Bad_Device, .CRD$GT_Which_CRD, CRD$GA_CRD_Message_Buffer); ![03]
;    0369  2     END;
;    0370  2
;    0371  2    [AUTO$K_Exit_AutoSizer_Error,          ![11]
;    0372  2     MENU$K_Exit_AutoSizer_Error]:         ![11]
;    0373  3     BEGIN
;    0374  3     IF (.DSA$V_CRD_Menutest_Off) THEN
;    0375  4      $CRD_Print (CRD$GT_CRD_EVSBA_Error, .CRD$GT_Which_CRD)     ![03]
;    0376  3     ELSE
;    0377  3      $CRD_Print (CRD$GT_CRD_EVSBB_Error, .CRD$GT_Which_CRD);    ![03]
;    0378  2     END;
;    0379  2    [AUTO$K_Exit_Inv_Base_System]: !        [17]
;    0380  3     BEGIN
;    0381  3     $CRD_Print (CRD$GT_Inv_Base_System);
;    0382  3     $CRD_Print (CRD$GT_CRD_Fatal_Error, .CRD$GT_Which_CRD);
;    0383  2     END;
;    0384  2    TES;
;    0385  2
;    0386  3    IF ((.DSA$V_CRD_AutoTest_Off) AND
;    0387  4     ((.Stop_Reason EQL AUTO$K_Exit_NoErrors_Complete)  OR
;    0388  4      (.Stop_Reason EQL AUTO$K_Exit_Errors_Complete)  OR
;    0389  4      (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Incomplete)  OR
;    0390  4      (.Stop_Reason EQL AUTO$K_Exit_Errors_Incomplete) OR
```

```
;    0391  5     (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Prep)
;    0392  4    )
;    0393  3      )
;    0394  2   THEN
;    0395  2    CRD$Auto_Summary ();   ! Get an Auto Test summary    ![17]
;    0396  2         ! only when exiting under specific circumstances
;    0397  2
;    0398  2   $CRD_Print (CRD$GT_CRD_End_Message, .CRD$GT_Which_CRD);       ![12]
;    0399  2
;    0400  4   IF (NOT (              ![14]
;    0401  4    (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Complete)   OR      ![14]
;    0402  4    (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Incomplete) OR      ![14]
;    0403  5    (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Prep)       ![14]
;    0404  2   )) THEN                ![14]
;    0405  2    Print_Exit_To ();           ![12]
;    0406  2
;    0407  2   IF (.DSA$V_CRD_AutoTest_Off) THEN         ![16]
;    0408  3    BEGIN              ![04]
;    0409  3    REGISTER             ![04]
;    0410  3     Next_Ptr,           ![14]
;    0411  3     DDB_Ptr : REF Device_Data_Block_Structure;      ![04]
;    0412  3
;    0413  3    !+                ![04]
;    0414  3    ! Deallocate all of the Device Data Blocks if DDB database exists.
;    0415  3    ! Use the CRD$Dispose routine to do this.  ![04]
;    0416  3    !-                ![04]
;    0417  3
;    0418  3    IF (DDB_Ptr = .CRD$GA_First_Diagnostic) NEQ 0      ! [17]
;    0419  3    THEN
;    0420  4        BEGIN
;    0421  4        DO              ![04]
;    0422  5     BEGIN             ![04]
;    0423  5     LOCAL             ![04]
;    0424  5      Size : BYTE;           ![14]
;    0425  5       ! To store the size of the DDB      ![04]
;    0426  5
;    0427  5     Size = .DDB_Ptr [CRD$K_DDB_Status] <DDB$B_Status_DDB_Size>;    ![06]
;    0428  5       ! Get the actual size which is stored in the DDB   ![04]
;    0429  5
;    0430  5     CRD$Print_DDB (.DDB_Ptr);
;    0431  5
;    0432  5     Next_Ptr = .DDB_Ptr [CRD$K_DDB_FLink];       ![14]
;    0433  5
;    0434  5     CRD$Dispose (.Size, .DDB_Ptr);        ![04]
;    0435  5       ! Dispose of the block of memory     ![04]
;    0436  5
;    0437  5     DDB_Ptr = .Next_Ptr;          ![14]
;    0438  5       ! Advance to the next DDB       .[04]
;    0439  5     END             ![04]
;    0440  4        UNTIL (.DDB_Ptr EQL .CRD$GA_First_Diagnostic);       ![04]
;    0441  3        END;
;    0442  3    END               ![07]
;    0443  2   ELSE IF (.DSA$V_CRD_MenuTest_Off) THEN      !  [16]
;    0444  3    BEGIN
;    0445  3    MEN$Menu_Cleanup ();          ![07]
```

```
;    0446  3    END                 ![11]
;    0447  2    ELSE IF (.DSA$V_CRD_MenuTest_On) THEN          ! [19]
;    0448  3    BEGIN                ! [19]
;    0449  3    MEN$Menu_Cleanup ();           ! [19]
;    0450  3    END                 ! [19]
;    0451  2    ELSE                 ![11]
; P  0452  2    $CRD_Print ($ASCIC (
; P  0453  2      '!/CRD INTERNAL ERROR - CRD$Stop - ',      ![15]
;    0454  2      'None of the CRD DSA bits are set. DSA=!XL hex!/'), .DSA$GL_Flags); ![15]
;    0455  2
;    0456  2    !+
;    0457  2    ! MENU TEST definitely occurs afters this point if stop_reason
;    0458  2    ! was _NoErrors_. BUT WHAT ABOUT _Errors_ ??
;    0459  2    !-
;    0460  3    IF ((.DSA$V_CRD_AutoTest_Off) AND
;    0461  4        ((.Stop_Reason EQL AUTO$K_Exit_NoErrors_Complete)  OR
;    0462  4         (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Incomplete)  OR
;    0463  5         (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Prep)
;    0464  4               )
;    0465  3        )
;    0466  2    THEN
;    0467  2     CRD$Screen_Pause (SCREEN$K_Press_Return_MM); ! Tell soft console operator to   ![17]
;    0468  2           !  Press Return for MAIN MENU...
;    0469  2           ! before continuing with Menu Test mode.
;    0470  2
;    0471  2    !+
;    0472  2    ! Finally, call (indirectly) the DSV$Exit routine. This will either exit to console mode
;    0473  2    ! (in stand-alone), or to VMS DCL (in user mode). This must be done LAST here because
;    0474  2    ! all the dispatch vectors are overwritten with debugging information. So, no $CRD_Print's
;    0475  2    ! can be done after this is done, etc.
;    0476  2    !-
;    0477  2
;    0478  4    IF (NOT (              ![14]
;    0479  4     (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Complete)   OR      ![14]
;    0480  4     (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Incomplete) OR      ![14]
;    0481  5     (.Stop_Reason EQL AUTO$K_Exit_NoErrors_Prep)       ![14]
;    0482  2    )) THEN              ![14]
;    0483  3     BEGIN
;    0484  3     LOCAL
;    0485  3      Exit_Address;
;    0486  3
;    0487  3     Exit_Address = .CRD$Exit_DS;
;    0488  3        ! Save this so that we can exit after the transfer
;    0489  3
; P  0490  3     CRD_Assert ((CRD$K_Total_Numb_Vectors EQL       ![15]
; P  0491  3      (CRD$K_Numb_UnUsed_Vectors + CRD$K_Numb_Dispatch_Vectors)),    ![15]
; L  0492  3      'Check the values of these constants in the CRD.R32 library');    ![15]
; %PRINT: ASSERT: The assertion is true.
;    0493  3
;    0494  3     !+
;    0495  3     ! Now transfer the debug vectors to the CRD's dispatch area. The DSR$UnLoad_CRD routine  ![15]
;    0496  3     ! will print out these vectors if an internal error has occurred. Note that    ![15]
;    0497  3     ! CRD_Dispatch_Vectors is actually mapped to the start of the actual vectors, not including  ![15]
;    0498  3     ! the version stuff in the first longword.        ![15]
;    0499  3     !-
```

```
;    0500  3
;    0501  3    INCR Vector FROM 0 TO (CRD$K_Total_Numb_Vectors - 1) DO      ![15]
;    0502  4    BEGIN
;    0503  4    CRD_Dispatch_Vectors [.Vector] = .CRD$GL_CRD_Debug_Vector [.Vector];
;    0504  3    END;
;    0505  3
;    0506  3    Jsb_None (.Exit_Address);
;    0507  3       ! JSB to the DSV$Exit routine
;    0508  2    END;
;    0509  2
;    0510  2    !+
;    0511  2    ! Continue on here if Auto Test is to go to Menu Test (automatically).     ![14]
;    0512  2    !-
;    0513  2
;    0514  2    DSA$V_CRD_AutoTest_Off = Off;          ![16]
;    0515  2    DSA$V_CRD_MenuTest_Off = On;          ![16]
;    0516  2    DSA$V_CRD_MenuTest_On = Off;          ![16]
;    0517  2    DSA$V_Binary       = On;          ![14]
;    0518  2    CRD$GB_Auto_To_Menu_Off = True; ! Indicate thet Menu is invoked
;    0519  2       ! from completion of Auto.      [16]
;    0520  2
;    0521  2    CRD$DS_Interface (CRD$K_Hook_Point, CRD$K_CRD_Initialization, 0, 0);     ![14]
;    0522  2
;    0523  2    JSB_Begin (.CRD$GA_Begin);          ![14]
;    0524  1 END;     ! Routine CRD$Stop


            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

4C  41  4E  52  45  54  4E  49  20  44  52  43  2F  21  51  00008 P.AAB:  .ASCII  \Q!/CRD INTERNAL ERROR - CRD$Stop - None \  ;
74  53  24  44  52  43  20  2D  20  52  4F  52  52  45  20  00017              ;
      20  65  6E  6F  4E  20  2D  20  70  6F  00026              ;
20  41  53  44  20  44  52  43  20  65  68  74  20  66  6F  00030    .ASCII  \of the CRD DSA bits are set. DSA=!XL hex\  ;
44  20  2E  74  65  73  20  65  72  61  20  73  74  69  62  0003F              ;
        78  65  68  20  4C  58  21  3D  41  53  0004E              ;
            2F  21  00058    .ASCII  \!/\            ;

            .EXTRN  CRD$DISABLE_CTRL_C
            .EXTRN  CRD$CLEAR_CTRL_C_FLAG
            .EXTRN  MEN$GT_SUPPORT_FILE_NOT_FOUND
            .EXTRN  MEN$GT_MENFUNCSUP_FILE
            .EXTRN  MEN$GT_SUPPORT_FILE_LOAD_ERR
            .EXTRN  CRD$GT_CONTROL_C_ABORT
            .EXTRN  CRD$GT_WHICH_CRD
            .EXTRN  CRD$GT_INTERNAL_ERROR
            .EXTRN  CRD$GT_INTERNAL_ERROR_ABORT
            .EXTRN  CRD$GT_CRD_NO_ERRORS
            .EXTRN  CRD$GT_CRD_ERRORS_COMPLETE
            .EXTRN  MEN$GT_OPERATOR_STOP
            .EXTRN  MEN$GT_OPERATOR_ABORT
            .EXTRN  CRD$GT_CRD_INCOMPLETE
            .EXTRN  CRD$GT_CRD_DEV_UNTESTED
            .EXTRN  CRD$GA_FIRST_DIAGNOSTIC
            .EXTRN  CRD$GA_CRD_MESSAGE_BUFFER
```

```
            .EXTRN  CRD$GT_SPACE_AND_SPACE
            .EXTRN  CRD$GT_CRD_BAD_DEVICE
            .EXTRN  CRD$GT_CRD_EVSBA_ERROR
            .EXTRN  CRD$GT_CRD_EVSBB_ERROR
            .EXTRN  CRD$GT_INV_BASE_SYSTEM
            .EXTRN  CRD$GT_CRD_FATAL_ERROR
            .EXTRN  CRD$GT_CRD_END_MESSAGE
            .EXTRN  CRD$PRINT_DDB, CRD$DISPOSE
            .EXTRN  MEN$MENU_CLEANUP
            .EXTRN  CRD$EXIT_DS, CRD$DS_INTERFACE
            .EXTRN  CRD$GA_BEGIN

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0FFC 00000    .ENTRY  CRD$STOP, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,-; 0184
             R11            ;
00000000G   EF   16 00002    JSB     CRD$DISABLE_CTRL_C          ; 0230
00000000G   EF   16 00008    JSB     CRD$CLEAR_CTRL_C_FLAG       ; 0231
    59 04    AC   D0 0000E    MOVL    STOP_REASON, R9         ; 0234
    0F       59   D1 00012    CMPL    R9, #15             ; 0236
    0F  12 00015    BNEQ    1$              ;
00000000G   EF   9F 00017    PUSHAB  MEN$GT_MENFUNCSUP_FILE      ; 0238
00000000G   EF   9F 0001D    PUSHAB  MEN$GT_SUPPORT_FILE_NOT_FOUND       ;
  01A9  31 00023    BRW     21$             ;
    10       59   D1 00026 1$:   CMPL    R9, #16             ; 0241
    0F  12 00029    BNEQ    2$              ;
00000000G   EF   9F 0002B    PUSHAB  MEN$GT_MENFUNCSUP_FILE      ; 0243
00000000G   EF   9F 00031    PUSHAB  MEN$GT_SUPPORT_FILE_LOAD_ERR        ;
  0195  31 00037    BRW     21$             ;
    59   D5 0003A 2$:   TSTL    R9              ; 0246
    0F  12 0003C    BNEQ    3$              ;
00000000G   EF   DD 0003E    PUSHL   CRD$GT_WHICH_CRD            ; 0248
00000000G   EF   9F 00044    PUSHAB  CRD$GT_CONTROL_C_ABORT          ;
  0182  31 0004A    BRW     21$             ;
    01       59   D1 0004D 3$:   CMPL    R9, #1              ; 0251
    05  13 00050    BEQL    4$              ;
    0D       59   D1 00052    CMPL    R9, #13             ;
    26  12 00055    BNEQ    5$              ;
00000000G   EF   DD 00057 4$:   PUSHL   CRD$GT_WHICH_CRD            ; 0254
00000000G   EF   9F 0005D    PUSHAB  CRD$GT_INTERNAL_ERROR           ;
    01   DD 00063    PUSHL   #1              ;
    1A   DD 00065    PUSHL   #26             ;
00000000G   FF       04  FB 00067    CALLS   #4, @CRD$PRINT          ;
00000000G   EF   DD 0006E    PUSHL   CRD$GT_WHICH_CRD            ; 0255
00000000G   EF   9F 00074    PUSHAB  CRD$GT_INTERNAL_ERROR_ABORT         ;
  0152  31 0007A    BRW     21$             ;
    05       59   D1 0007D 5$:   CMPL    R9, #5              ; 0258
    0F  12 00080    BNEQ    6$              ;
00000000G   EF   DD 00082    PUSHL   CRD$GT_WHICH_CRD            ; 0260
00000000G   EF   9F 00088    PUSHAB  CRD$GT_CRD_NO_ERRORS            ;
  013E  31 0008E    BRW     21$             ;
    03       59   D1 00091 6$:   CMPL    R9, #3              ; 0263
    0F  12 00094    BNEQ    7$              ;
00000000G   EF   DD 00096    PUSHL   CRD$GT_WHICH_CRD            ; 0265
00000000G   EF   9F 0009C    PUSHAB  CRD$GT_CRD_ERRORS_COMPLETE          ;
```

```
        012A  31 000A2     BRW     21$             ;
          0B      59  D1 000A5 7$:   CMPL    R9, #11          ; 0268
          08  12 000A8     BNEQ    8$              ;
    00000000G  EF  9F 000AA     PUSHAB  MEN$GT_OPERATOR_STOP        ; 0270
          0B  11 000B0     BRB     9$              ;
          0C      59  D1 000B2 8$:   CMPL    R9, #12          ; 0273
          14  12 000B5     BNEQ    10$             ;
    00000000G  EF  9F 000B7     PUSHAB  MEN$GT_OPERATOR_ABORT        ; 0284
          01  DD 000BD 9$:   PUSHL   #1              ;
          1A  DD 000BF     PUSHL   #26             ;
    00000000G  FF      03  FB 000C1     CALLS   #3, @CRD$PRINT          ;
        010F  31 000C8     BRW     22$             ; 0234
          06      59  D1 000CB 10$:   CMPL    R9, #6           ; 0287
          0F  12 000CE     BNEQ    11$             ;
    00000000G  EF  DD 000D0     PUSHL   CRD$GT_WHICH_CRD         ; 0289
    00000000G  EF  9F 000D6     PUSHAB  CRD$GT_CRD_INCOMPLETE        ;
        00F0  31 000DC     BRW     21$             ;
          07      59  D1 000DF 11$:   CMPL    R9, #7           ; 0292
          0F  12 000E2     BNEQ    12$             ;
    00000000G  EF  DD 000E4     PUSHL   CRD$GT_WHICH_CRD         ; 0294
    00000000G  EF  9F 000EA     PUSHAB  CRD$GT_CRD_DEV_UNTESTED      ;
        00DC  31 000F0     BRW     21$             ;
          04      59  D1 000F3 12$:   CMPL    R9, #4           ; 0297
          03  13 000F6     BEQL    13$             ;
        0085  31 000F8     BRW     17$             ;
          58 00000000G  EF  9E 000FB 13$:   MOVAB   CRD$GA_CRD_MESSAGE_BUFFER+1, CHAR_PTR      ; 0313
          5A  D4 00102     CLRL    NUMB_BAD_DEVICES        ; 0314
          56 00000000G  EF  D0 00104     MOVL    CRD$GA_FIRST_DIAGNOSTIC, DDB_PTR      ; 0315
          36  14  A6      04  E1 0010B 14$:   BBC     #4, 20(DDB_PTR), 16$        ; 0319
          50 0C   A6  D0 00110     MOVL    12(DDB_PTR), R0         ; 0322
          57 0C   A0  D0 00114     MOVL    12(R0), R7          ; 0326
          5A  D6 00118     INCL    NUMB_BAD_DEVICES        ; 0329
          01      5A  D1 0011A     CMPL    NUMB_BAD_DEVICES, #1        ; 0335
          19  15 0011D     BLEQ    15$             ;
          50 00000000G  EF  9A 0011F     MOVZBL  CRD$GT_SPACE_AND_SPACE, R0        ; 0341
          68 00000000G  EF      50  28 00126     MOVC3   R0, CRD$GT_SPACE_AND_SPACE+1, (CHAR_PTR)    ; 0343
          50 00000000G  EF  9A 0012E     MOVZBL  CRD$GT_SPACE_AND_SPACE, R0        ; 0345
          58      50  C0 00135     ADDL2   R0, CHAR_PTR            ;
          50      67  9A 00138 15$:   MOVZBL  (R7), R0            ; 0353
          68   01   A7      50  28 0013B     MOVC3   R0, 1(R7), (CHAR_PTR)         ; 0355
          50      67  9A 00140     MOVZBL  (R7), R0            ; 0357
          58      50  C0 00143     ADDL2   R0, CHAR_PTR            ;
          56      66  D0 00146 16$:   MOVL    (DDB_PTR), DDB_PTR          ; 0360
    00000000G  EF      56  D1 00149     CMPL    DDB_PTR, CRD$GA_FIRST_DIAGNOSTIC      ; 0363
          B9  12 00150     BNEQ    14$             ;
          50 00000000G  EF  9E 00152     MOVAB   CRD$GA_CRD_MESSAGE_BUFFER+1, R0       ; 0366
    00000000G  EF      58      50  83 00159     SUBB3   R0, CHAR_PTR, CRD$GA_CRD_MESSAGE_BUFFER     ;
    00000000G  EF  9F 00161     PUSHAB  CRD$GA_CRD_MESSAGE_BUFFER       ; 0368
    00000000G  EF  DD 00167     PUSHL   CRD$GT_WHICH_CRD            ;
    00000000G  EF  9F 0016D     PUSHAB  CRD$GT_CRD_BAD_DEVICE           ;
          01  DD 00173     PUSHL   #1              ;
          1A  DD 00175     PUSHL   #26             ;
    00000000G  FF      05  FB 00177     CALLS   #5, @CRD$PRINT          ;
          5A  11 0017E     BRB     22$             ; 0234
          02      59  D1 00180 17$:   CMPL    R9, #2           ; 0371
```

```
            05  13 00183     BEQL    18$             ;
            0E        59  D1 00185     CMPL    R9, #14         ;
            23  12 00188     BNEQ    20$             ;
            0E 0000FE02   9F  E9 0018A 18$:   BLBC    @#^X0000FE02, 19$         ; 0374
      00000000G  EF  DD 00191     PUSHL   CRD$GT_WHICH_CRD          ; 0375
      00000000G  EF  9F 00197     PUSHAB  CRD$GT_CRD_EVSBA_ERROR        ;
            30  11 0019D     BRB     21$             ;
      00000000G  EF  DD 0019F 19$:   PUSHL   CRD$GT_WHICH_CRD          ; 0377
      00000000G  EF  9F 001A5     PUSHAB  CRD$GT_CRD_EVSBB_ERROR        ;
            22  11 001AB     BRB     21$             ;
            08        59  D1 001AD 20$:   CMPL    R9, #8          ; 0379
            28  12 001B0     BNEQ    22$             ;
      00000000G  EF  9F 001B2     PUSHAB  CRD$GT_INV_BASE_SYSTEM        ; 0381
            01  DD 001B8     PUSHL   #1              ;
            1A  DD 001BA     PUSHL   #26             ;
      00000000G  FF        03  FB 001BC     CALLS   #3, @CRD$PRINT          ;
      00000000G  EF  DD 001C3     PUSHL   CRD$GT_WHICH_CRD          ; 0382
      00000000G  EF  9F 001C9     PUSHAB  CRD$GT_CRD_FATAL_ERROR        ;
            01  DD 001CF 21$:   PUSHL   #1              ;
            1A  DD 001D1     PUSHL   #26             ;
      00000000G  FF        04  FB 001D3     CALLS   #4, @CRD$PRINT          ;
   57 0000FE01   9F        01      03  EF 001DA 22$:   EXTZV   #3, #1, @#^X0000FE01, R7        ; 0386
            20        57  E9 001E3     BLBC    R7, 24$         ;
            05        59  D1 001E6     CMPL    R9, #5          ; 0387
            14  13 001E9     BEQL    23$             ;
            03        59  D1 001EB     CMPL    R9, #3          ; 0388
            0F  13 001EE     BEQL    23$             ;
            06        59  D1 001F0     CMPL    R9, #6          ; 0389
            0A  13 001F3     BEQL    23$             ;
            04        59  D1 001F5     CMPL    R9, #4          ; 0390
            05  13 001F8     BEQL    23$             ;
            07        59  D1 001FA     CMPL    R9, #7          ; 0391
            07  12 001FD     BNEQ    24$             ;
      00000000V  EF        00  FB 001FF 23$:   CALLS   #0, CRD$AUTO_SUMMARY          ; 0395
      00000000G  EF  DD 00206 24$:   PUSHL   CRD$GT_WHICH_CRD          ; 0398
      00000000G  EF  9F 0020C     PUSHAB  CRD$GT_CRD_END_MESSAGE        ;
            01  DD 00212     PUSHL   #1              ;
            1A  DD 00214     PUSHL   #26             ;
      00000000G  FF        04  FB 00216     CALLS   #4, @CRD$PRINT          ;
            56  D4 0021D     CLRL    R6              ; 0401
            05        59  D1 0021F     CMPL    R9, #5          ;
            02  12 00222     BNEQ    25$             ;
            56  D6 00224     INCL    R6              ;
            50  D4 00226 25$:   CLRL    R0              ; 0402
            06        59  D1 00228     CMPL    R9, #6          ;
            02  12 0022B     BNEQ    26$             ;
            50  D6 0022D     INCL    R0              ;
            50        56  C8 0022F 26$:   BISL2   R6, R0          ;
            55  D4 00232     CLRL    R5              ; 0403
            07        59  D1 00234     CMPL    R9, #7          ;
            02  12 00237     BNEQ    27$             ;
            55  D6 00239     INCL    R5              ;
            55        50  C8 0023B 27$:   BISL2   R0, R5          ;
            05        55  E8 0023E     BLBS    R5, 28$           ; 0400
   FD7A     CF        00  FB 00241     CALLS   #0, PRINT_EXIT_TO         ; 0405
```

```
            33      57  E9 00246 28$:   BLBC    R7, 30$                 ; 0407
            53 00000000G  EF  D0 00249    MOVL    CRD$GA_FIRST_DIAGNOSTIC, DDB_PTR        ; 0418
            59  13 00250    BEQL    33$             ;
            54 17   A3  90 00252 29$:   MOVB    23(DDB_PTR), SIZE           ; 0427
            53  DD 00256    PUSHL   DDB_PTR             ; 0430
        00000000G  EF       01  FB 00258    CALLS   #1, CRD$PRINT_DDB           ;
            52      63  D0 0025F    MOVL    (DDB_PTR), NEXT_PTR         ; 0432
            53  DD 00262    PUSHL   DDB_PTR             ; 0434
            7E      54  9A 00264    MOVZBL  SIZE, -(SP)             ;
        00000000G  EF       02  FB 00267    CALLS   #2, CRD$DISPOSE             ;
            53      52  D0 0026E    MOVL    NEXT_PTR, DDB_PTR           ; 0437
        00000000G  EF       53  D1 00271    CMPL    DDB_PTR, CRD$GA_FIRST_DIAGNOSTIC        ; 0440
            D8  12 00278    BNEQ    29$             ;
            2F  11 0027A    BRB     33$             ; 0407
            08 0000FE02   9F  E8 0027C 30$:   BLBS    @#^X0000FE02, 31$           ; 0443
            09 0000FE02   9F      02  E1 00283    BBC     #2, @#^X0000FE02, 32$           ; 0447
        00000000G  EF       00  FB 0028B 31$:   CALLS   #0, MEN$MENU_CLEANUP            ; 0449
            17  11 00292    BRB     33$             ; 0447
        0000FE00   9F  DD 00294 32$:   PUSHL   @#^X0000FE00            ; 0454
        00000000'  EF  9F 0029A    PUSHAB  P.AAB               ;
            01  DD 002A0    PUSHL   #1              ;
            1A  DD 002A2    PUSHL   #26             ;
        00000000G  FF       04  FB 002A4    CALLS   #4, @CRD$PRINT              ;
            16      57  E9 002AB 33$:   BLBC    R7, 35$                 ; 0460
            0A      56  E8 002AE    BLBS    R6, 34$             ; 0461
            06      59  D1 002B1    CMPL    R9, #6              ; 0462
            05  13 002B4    BEQL    34$             ;
            07      59  D1 002B6    CMPL    R9, #7              ; 0463
            09  12 002B9    BNEQ    35$             ;
            03  DD 002BB 34$:   PUSHL   #3              ; 0467
        00000000V  EF       01  FB 002BD    CALLS   #1, CRD$SCREEN_PAUSE            ;
            1C      55  E8 002C4 35$:   BLBS    R5, 37$             ; 0478
            51 00000000G  EF  D0 002C7    MOVL    CRD$EXIT_DS, EXIT_ADDRESS           ; 0487
            50  D4 002CE    CLRL    VECTOR              ; 0501
        00000000GEF40 00000000'EF40  D0 002D0 36$:   MOVL    CRD$GL_CRD_DEBUG_VECTOR[VECTOR], -          ; 0503
             CRD_DISPATCH_VECTORS[VECTOR]           ;
            EF      50      18  F3 002DD    AOBLEQ  #24, VECTOR, 36$            ; 0501
            61  16 002E1    JSB     (EXIT_ADDRESS)              ; 0506
        0000FE01   9F       08  8A 002E3 37$:   BICB2   #8, @#^X0000FE01            ; 0514
        0000FE02   9F       01  88 002EA    BISB2   #1, @#^X0000FE02            ; 0515
        0000FE02   9F       04  8A 002F1    BICB2   #4, @#^X0000FE02            ; 0516
        0000FE00   9F       02  88 002F8    BISB2   #2, @#^X0000FE00            ; 0517
        00000000'  EF       01  90 002FF    MOVB    #1, CRD$GB_AUTO_TO_MENU_OFF         ; 0518
            7E  7C 00306    CLRQ    -(SP)               ; 0521
            7E  7C 00308    CLRQ    -(SP)               ;
        00000000G  EF       04  FB 0030A    CALLS   #4, CRD$DS_INTERFACE            ;
        00000000G  FF  16 00311    JSB     @CRD$GA_BEGIN               ; 0523
         04 00317    RET                 ; 0524

; Routine Size:  792 bytes,    Routine Base:  CODE + 0040
```

```
;    0525  1 %SBTTL 'CRD$Control_C_Handler Routine'
;    0526  1
;    0527  1 GLOBAL ROUTINE CRD$Control_C_Handler : NOVALUE =
;    0528  1
;    0529  1 !++
;    0530  1 ! Functional Description:
;    0531  1 !
;    0532  1 ! Handle Control-C's during CRD.
;    0533  1 !
;    0534  1 ! Formal Input Parameters:
;    0535  1 !
;    0536  1 ! None
;    0537  1 !
;    0538  1 ! Formal Output Parameters:
;    0539  1 !
;    0540  1 ! None
;    0541  1 !
;    0542  1 ! Side Effects:
;    0543  1 !
;    0544  1 ! None
;    0545  1 !
;    0546  1 ! Completion Codes:
;    0547  1 !
;    0548  1 ! None
;    0549  1 !-
;    0550  2 BEGIN     ! Routine CRD$Control_C_Handler
;    0551  2  CRD$Disable_Ctrl_C ();              ![12]
;    0552  2  CRD$Clear_Ctrl_C_Flag ();           ![12]
;    0553  2
;    0554  2  CRD$Stop (AUTO$K_Exit_Control_C);         ![11]
;    0555  2      ! Call Stop because of the Control-C
;    0556  1 END;     ! Routine CRD$Control_C_Handler




          0FFC 00000     .ENTRY  CRD$CONTROL_C_HANDLER, Save R2,R3,R4,R5,R6,-; 0527
         R7,R8,R9,R10,R11          ;
    00000000G  EF  16 00002     JSB     CRD$DISABLE_CTRL_C          ; 0551
    00000000G  EF  16 00008     JSB     CRD$CLEAR_CTRL_C_FLAG          ; 0552
        7E  D4 0000E     CLRL    -(SP)               ; 0554
FCD3    CF      01  FB 00010     CALLS   #1, CRD$STOP          ;
     04 00015     RET                 ; 0556

; Routine Size:  22 bytes,    Routine Base:  CODE + 0358


;    0557  1
;    0558  1
```

```
;    0559  1 %SBTTL 'CRD$Auto_Summary Routine'
;    0560  1
;    0561  1 GLOBAL ROUTINE CRD$Auto_Summary : NOVALUE =
;    0562  1
;    0563  1 !++
;    0564  1 ! Functional Description:
;    0565  1 !
;    0566  1 ! Prints a summary of devices tested and their status after the auto test
;    0567  1 ! only for soft consoles. Otherwise, just return.
;    0568  1 !
;    0569  1 ! Formal Input Parameters:
;    0570  1 !
;    0571  1 ! None
;    0572  1 !
;    0573  1 ! Formal Output Parameters:
;    0574  1 !
;    0575  1 ! None
;    0576  1 !
;    0577  1 ! Side Effects:
;    0578  1 !
;    0579  1 ! None
;    0580  1 !
;    0581  1 ! Completion Codes:
;    0582  1 !
;    0583  1 ! None
;    0584  1 !--
;    0585  2 BEGIN     ! Routine CRD$Auto_Summary
;    0586  2     LOCAL
;    0587  2  Some_Fail: INITIAL (False),
;    0588  2  ! CPU and MEMORY always pass in microdiagnostics before Auto Summary
;    0589  2  ! Some_Pass: INITIAL (False),
;    0590  2  Some_Inc_Prep: INITIAL (False),
;    0591  2  Some_Other: INITIAL (False),
;    0592  2  Console_Char : $CRD_Termchar_ATTR;
;    0593  2
;    0594  2     REGISTER
;    0595  2  Hdwr_Name_Ptr,
;    0596  2  Dev_Name,
;    0597  2  Screen_Counter,
;    0598  2  DDB_Ptr  : REF Device_Data_Block_Structure,
;    0599  2  SupBlk_Ptr : REF $MEN_SupBlk_ATTR,
;    0600  2  PTable_Ptr : REF $DS_HPO_DECL ();
;    0601  2
;    0602  2     MAP
;    0603  2  CRD$GA_First_Diagnostic  : REF Device_Data_Block_Structure;
;    0604  2
;    0605  2
;    0606  2     !+
;    0607  2     ! BEFORE DOING ANYTHING, CHECK THAT THE CONSOLE IS SOFT (i.e. VT100 or VT52)
;    0608  2     ! AND NOT HARD (i.e. LA34 etc.)
;    0609  2     !-
;    0610  3     IF ($DS_GETTERM (TERMCHAR = Console_Char))
;    0611  3      ! If SS$_NORMAL status on getting
;    0612  3      ! terminal characteristics
;    0613  2     THEN
```

```
;    0614  3       BEGIN
;    0615  3       !+
;    0616  3  ! If console is not a CRD supported soft console, then just RETURN.
;    0617  3  ! Otherwise, continue normally.
;    0618  3  !-
;    0619  4  IF ((.Console_Char [Termchar$B_Type] NEQ TT$_VT100) AND
;    0620  4 !***     (.Console_Char [Termchar$B_Type] NEQ TT$_VS125) AND
;    0621  4       (.Console_Char [Termchar$B_Type] NEQ TT$_VT52)  AND
;    0622  5       (.Console_Char [Termchar$B_Type] NEQ TT$_VT200_Series)
;    0623  4     )
;    0624  3  THEN
;    0625  3   RETURN;
;    0626  3       END
;    0627  2     ELSE
;    0628  2      $CRD_Print (CRD$GT_Internal_Error);
;    0629  2
;    0630  2
;    0631  2      !+
;    0632  2      ! Print the Auto Test Summary Title
;    0633  2      !-
;    0634  2
;    0635  2      $CRD_Print (CRD$GT_Summary_Title);
;    0636  2
;    0637  2
;    0638  2      INCR I FROM 0 TO 4 DO
;    0639  3  BEGIN    ! Begin of INCR I
;    0640  3  !+
;    0641  3  ! Print the Subtitle of the Auto Test Summary
;    0642  3  !-
;    0643  3  SELECTONE .I OF
;    0644  3   SET
;    0645  3   [1]: IF (.Some_Fail) THEN $CRD_Print (CRD$GT_All_Failures);
;    0646  3
;    0647  3        ! CPU and MEMORY always pass in microdiagnostics to get to this stage
;    0648  3   [2]: $CRD_Print (CRD$GT_All_Passes);
;    0649  3
;    0650  3   [3]: IF (.Some_Inc_Prep) THEN $CRD_Print (CRD$GT_All_Inc_Hdwr_Prep);
;    0651  3   [4]: IF (.Some_Other) THEN $CRD_Print (CRD$GT_All_Others);
;    0652  3   TES;
;    0653  3
;    0654  3  DDB_Ptr = .CRD$GA_First_Diagnostic;
;    0655  3      . Set DDB_Ptr to point to the first DDB
;    0656  3
;    0657  3  Screen_Counter - 0;  ! Init number of entries actually on
;    0658  3      ! 1 line of the screen. Max number is 3
;    0659  3
;    0660  3  !+
;    0661  3  ! CPU and MEMORY always pass in microdiagnostics to get to this stage
;    0662  3  !-
;    0663  4  IF (.I EQL 2)
;    0664  3  THEN
;    0665  4   BEGIN
;    0666  4      $CRD_Print (CRD$GT_Init_All_Passes);
;    0667  4      Screen_Counter = 2;
;    0668  3   END;
```

```
;    0669  3
;    0670  3  !+
;    0671  3  ! The following DO-UNTIL acts on each DDB in the DDBs-queue.
;    0672  3  !-
;    0673  3
;    0674  3  DO
;    0675  3
;    0676  3   !+
;    0677  3   ! This is a large DO in the DO_UNTIL command.
;    0678  3   !-
;    0679  4   BEGIN   ! Begin of DO command.
;    0680  4
;    0681  4   !+
;    0682  4   ! For this DDB:
;    0683  4   ! 1st: determine whether device test failed, passed or was inc prep hdwr
;    0684  4   ! 2nd: print the DDB info if device test failed and we are printing the list
;    0685  4   !  of failed devices
;    0686  4   ! last: move on to the next DDB.
;    0687  4   !
;    0688  4   ! NOTES ON FAIL vs PASS vs INC HDWR PREP:
;    0689  4   ! FAIL is only for essential devices.
;    0690  4   ! INC HDWR PREP, as far as the summary is concerned, is only
;    0691  4   !  for essential devices and for non-essential devices which
;    0692  4   !  are also availaable.
;    0693  4   ! PASS is the whatever is not FAIL and not INC HDWR PREP BUT
;    0694  4   !  non-essential devices which are also not available
;    0695  4   !  will not be in this list nor in any Auto Summary list.
;    0696  4   !-
;    0697  4   REGISTER
;    0698  4    Prep_Error  : BYTE,
;    0699  4    Bad   : BYTE,
;    0700  4    Essential_Device : BYTE,
;    0701  4    Finished  : BYTE,
;    0702  4    DDB_Fail : BYTE,
;    0703  4    DDB_Other : BYTE,
;    0704  4    DDB_Pass : BYTE,
;    0705  4    DDB_Inc_Prep : BYTE;
;    0706  4
;    0707  4   Finished  = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete>;
;    0708  4   Essential_Device = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Essential_Device>;
;    0709  4   Bad   = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Device_Bad>;
;    0710  4   Prep_Error  = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Preparation_Error>;
;    0711  4
;    0712  4
;    0713  5   IF (.DDB_Ptr [CRD$K_DDB_PTable_Entry] NEQ 0)
;    0714  4   THEN
;    0715  5      BEGIN
;    0716  5    !+
;    0717  5    ! Determine if DDB is FAIL
;    0718  5    !-
;    0719  7    DDB_Fail = (  ((NOT(.Finished)) OR .Bad)
;    0720  6      AND
;    0721  7      (NOT (.Prep_Error))
;    0722  5           );
;    0723  5    !+
```

```
;    0724  5     ! Determine if Incorrect Preparation
;    0725  5     !-
;    0726  5     DDB_Inc_Prep = Boolean (.Prep_Error);
;    0727  5
;    0728  5     !+
;    0729  5     ! Determine if DDB is PASS
;    0730  5     !-
;    0731  5     DDB_Pass = ((NOT(.DDB_Fail))  AND  (NOT(.DDB_Inc_Prep)));
;    0732  5
;    0733  5     !+
;    0734  5     ! Determine if DDB is OTHERS
;    0735  5     !-
;    0736  5     DDB_Other = False;
;    0737  4       END;
;    0738  4
;    0739  4
;    0740  5    IF (.DDB_Ptr [CRD$K_DDB_PTable_Entry] EQL 0)
;    0741  4    THEN
;    0742  5       BEGIN
;    0743  5     DDB_Fail = .Essential_Device;
;    0744  5     DDB_Inc_Prep = False;
;    0745  5     DDB_Pass = False;
;    0746  5     DDB_Other = NOT (.Essential_Device);
;    0747  4       END;
;    0748  4
;    0749  4    !+
;    0750  4    ! Determine if any DDB thus far has FAILed or INC PREPed
;    0751  4    ! Note there is no Some_Pass because CPU and MEMORY always
;    0752  4    ! pass in microdiagnostics before Auto Summary
;    0753  4    !-
;    0754  4    Some_Fail = .Some_Fail OR .DDB_Fail;
;    0755  4
;    0756  4    Some_Inc_Prep = .Some_Inc_Prep  OR .DDB_Inc_Prep;
;    0757  4
;    0758  4    Some_Other = .Some_Other OR .DDB_Other;
;    0759  4
;    0760  5    IF (SELECTONE .I OF
;    0761  5     SET
;    0762  5     [0]: False; ! Do no printing for this pass thru DDBs
;    0763  5     [1]: .DDB_Fail;
;    0764  5     [2]: .DDB_Pass;
;    0765  5     [3]: .DDB_Inc_Prep;
;    0766  5     [4]: .DDB_Other;
;    0767  5     TES
;    0768  5       )    ! End of SELECTONE statement
;    0769  4    THEN
;    0770  5         BEGIN
;    0771  5     !+
;    0772  5     ! Set up for printing Hardware Name
;    0773  5     !-
;    0774  5
;    0775  5     PTable_Ptr = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    0776  5       ! Point to the P-Table
;    0777  5     !+
;    0778  5     ! If there is a P-Table for this device, then
```

```
;    0779  5     ! use it to get the generic name
;    0780  5     ! Otherwise, we only know what the device name
;    0781  5     ! should be.
;    0782  5     !-
;    0783  6     IF (.PTable_Ptr NEQ 0)
;    0784  5     THEN
;    0785  6        BEGIN
;    0786  6      Hdwr_Name_Ptr = PTable_Ptr [HP$T_TYPE];
;    0787  6       ! Point to the Hardware device name ASCIC
;    0788  6
;    0789  6      !+
;    0790  6      ! Set up for printing Generic Device Name
;    0791  6      !-
;    0792  6
;    0793  6      Dev_Name = CRD$Get_Device_Name (.PTable_Ptr);
;    0794  6       ! Get an ASCIC device name string from the Ptable
;    0795  6        END
;    0796  5     ELSE
;    0797  6        BEGIN
;    0798  6         BIND Diag_Support_Vector = (.DDB_Ptr [CRD$K_DDB_Auto_Support_Entry]) : Hardware_Support_Vector;
;    0799  6      LOCAL
;    0800  6       Generic_Dev_Name : REF VECTOR [,BYTE],
;    0801  6       Temp_Dev_Name : REF VECTOR [,BYTE];
;    0802  6
;    0803  6         Dev_Name = .Diag_Support_Vector [CRD$K_Device_Name];
;    0804  6         Generic_Dev_Name = .Diag_Support_Vector [CRD$K_Device_Name];
;    0805  6
;    0806  6         INCR J FROM 0 TO 7 DO
;    0807  7            BEGIN
;    0808  7       ! Start INCR-DO loop
;    0809  7          SELECTONE .J OF
;    0810  7             SET
;    0811  8             [0]: BEGIN
;    0812  8              Temp_Dev_Name = $ASCIC ('DQA1');
;    0813  8              Hdwr_Name_Ptr = $ASCIC ('RL02');
;    0814  7           END;
;    0815  8             [1]: BEGIN
;    0816  8              Temp_Dev_Name = $ASCIC ('XGA0');
;    0817  8              Hdwr_Name_Ptr = $ASCIC ('DMF32S');
;    0818  7           END;
;    0819  8             [2]: BEGIN
;    0820  8              Temp_Dev_Name = $ASCIC ('TXA');
;    0821  8              Hdwr_Name_Ptr = $ASCIC ('DMF32A');
;    0822  7           END;
;    0823  8             [3]: BEGIN
;    0824  8              Temp_Dev_Name = $ASCIC ('LCA');
;    0825  8              Hdwr_Name_Ptr = $ASCIC ('DMF32P');
;    0826  7           END;
;    0827  8             [4]: BEGIN
;    0828  8              Temp_Dev_Name = $ASCIC ('DAA0');
;    0829  8              Hdwr_Name_Ptr = $ASCIC ('RC25');
;    0830  7           END;
;    0831  8             [5]: BEGIN
;    0832  8              Temp_Dev_Name = $ASCIC ('DAA1');
;    0833  8              Hdwr_Name_Ptr = $ASCIC ('RCF25');
```

```
;    0834  7          END;
;    0835  8            [6]: BEGIN
;    0836  8             Temp_Dev_Name = $ASCIC ('DJA1');
;    0837  8             Hdwr_Name_Ptr = $ASCIC ('RA60');
;    0838  7          END;
;    0839  8            [7]: BEGIN
;    0840  8             Temp_Dev_Name = $ASCIC ('DJA0');
;    0841  8             Hdwr_Name_Ptr = $ASCIC ('RA60');
;    0842  7          END;
;    0843  8         [OTHERWISE]: BEGIN
;    0844  8           Hdwr_Name_Ptr = crd$gt_unknown;
;    0845  7        END;
;    0846  7            TES;
;    0847  7
;    0848  8         IF (CH$EQL (.Generic_Dev_Name [0], Generic_Dev_Name [1],
;    0849  8           .Temp_Dev_Name [0], Temp_Dev_Name [1])
;    0850  8            )
;    0851  7         THEN
;    0852  7            EXITLOOP;
;    0853  6           END; ! End of INCR-DO loop
;    0854  5       END;  ! End of P_Table EQL 0
;    0855  5
;    0856  5
;    0857  5
;    0858  5     !+
;    0859  5     ! Print the system hardware text
;    0860  5     !-
;    0861  5
;    0862  5     !+
;    0863  5     ! For the Other category, the format is different
;    0864  5     ! than for the Pass, Fail, or Inc Hdwr Prep.
;    0865  5     !-
;    0866  5
;    0867  6     IF (.I NEQ 4)
;    0868  5     THEN
;    0869  6          BEGIN
;    0870  7      IF (.Screen_Counter EQL 3)
;    0871  6      THEN
;    0872  7       BEGIN
;    0873  7       $CRD_Print (CRD$GT_New_Line);
;    0874  7       Screen_Counter = 0;
;    0875  6       END;
;    0876  6
;    0877  6      Screen_Counter = .Screen_Counter + 1;
;    0878  5          END;
;    0879  5
;    0880  5     $CRD_Print (CRD$GT_Tab);
;    0881  5     $CRD_Print (.Hdwr_Name_Ptr);
;    0882  5     $CRD_Print (CRD$GT_Slash);
;    0883  5     $CRD_Print (.Dev_Name);
;    0884  5     $CRD_Print (CRD$GT_Tab);
;    0885  5
;    0886  6     IF (.I EQL 4)
;    0887  5     THEN
;    0888  5      $CRD_Print (CRD$GT_Dev_Unavailable_2);
```

```
;    0889  5
;    0890  5         END
;    0891  4    ;    ! End of large IF THEN command
;    0892  4
;    0893  4
;    0894  4    !+
;    0895  4    ! We are done processing this DDB so move on to the next DDB
;    0896  4    !-
;    0897  4
;    0898  4    DDB_Ptr = .DDB_Ptr [CRD$K_DDB_Flink];
;    0899  4
;    0900  4    END    ! End of DO command in DO-UNTIL loop.
;    0901  4
;    0902  3   UNTIL (.DDB_Ptr EQL .CRD$GA_First_Diagnostic);
;    0903  3       ! End of UNTIL-DO loop.
;    0904  3
;    0905  5      IF (  (NOT (.Some_Other))  ! Make sure there is always a space
;    0906  4         AND     ! between summary and END of AUTO
;    0907  5     (.I EQL 4)    ! message.
;    0908  4         )
;    0909  3      THEN
;    0910  3   $CRD_Print (CRD$GT_New_Line);
;    0911  3
;    0912  3
;    0913  2      END;     ! End of INCR I FROM 0 TO 4 loop.
;    0914  1 END;     ! End CRD$Auto_Summary



          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

 31  41  51  44  04   0005A P.AAC:   .ASCII  <4>\DQA1\           ;
 32  30  4C  52  04   0005F P.AAD:   .ASCII  <4>\RL02\           ;
 30  41  47  58  04   00064 P.AAE:   .ASCII  <4>\XGA0\           ;
53  32  33  46  4D  44  06   00069 P.AAF:   .ASCII  <6>\DMF32S\          ;
     41  58  54  03   00070 P.AAG:   .ASCII  <3>\TXA\          ;
41  32  33  46  4D  44  06   00074 P.AAH:   .ASCII  <6>\DMF32A\          ;
     41  43  4C  03   0007B P.AAI:   .ASCII  <3>\LCA\          ;
50  32  33  46  4D  44  06   0007F P.AAJ:   .ASCII  <6>\DMF32P\          ;
 30  41  41  44  04   00086 P.AAK:   .ASCII  <4>\DAA0\           ;
 35  32  43  52  04   0008B P.AAL:   .ASCII  <4>\RC25\           ;
 31  41  41  44  04   00090 P.AAM:   .ASCII  <4>\DAA1\           ;
    35  32  46  43  52  05   00095 P.AAN:   .ASCII  <5>\RCF25\          ;
 31  41  4A  44  04   0009B P.AAO:   .ASCII  <4>\DJA1\           ;
 30  36  41  52  04   000A0 P.AAP:   .ASCII  <4>\RA60\           ;
 30  41  4A  44  04   000A5 P.AAQ:   .ASCII  <4>\DJA0\           ;
 30  36  41  52  04   000AA P.AAR:   .ASCII  <4>\RA60\           ;

        .EXTRN  DS$GETTERM, CRD$GT_SUMMARY_TITLE
        .EXTRN  CRD$GT_ALL_FAILURES
        .EXTRN  CRD$GT_ALL_PASSES
        .EXTRN  CRD$GT_ALL_INC_HDWR_PREP
        .EXTRN  CRD$GT_ALL_OTHERS
        .EXTRN  CRD$GT_INIT_ALL_PASSES
        .EXTRN  CRD$GET_DEVICE_NAME
```

```
                .EXTRN  CRD$GT_UNKNOWN, CRD$GT_NEW_LINE
                .EXTRN  CRD$GT_TAB, CRD$GT_SLASH
                .EXTRN  CRD$GT_DEV_UNAVAILABLE_2

                .PSECT  CODE,NOWRT,  SHR,  PIC,2

                  0FFC 00000      .ENTRY  CRD$AUTO_SUMMARY, Save R2,R3,R4,R5,R6,R7,-   ; 0561
                 R8,R9,R10,R11            ;
                5E       0C  C2 00002     SUBL2   #12, SP          ;
                7E   7C 00005     CLRQ     SOME_INC_PREP          ; 0585
                7E   D4 00007     CLRL     SOME_OTHER           ;
            10  AE   9F 00009     PUSHAB   CONSOLE_CHAR           ; 0610
             00000000G  9F       01  FB 0000C     CALLS   #1, a#DS$GETTERM          ;
                16       50  E9 00013     BLBC    R0, 1$           ;
            60  8F 11    AE  91 00016     CMPB    CONSOLE_CHAR+1, #96          ; 0619
                20   13 0001B     BEQL    2$               ;
            40  8F 11    AE  91 0001D     CMPB    CONSOLE_CHAR+1, #64          ; 0621
                19   13 00022     BEQL    2$               ;
            6E  8F 11    AE  91 00024     CMPB    CONSOLE_CHAR+1, #110          ; 0622
                12   13 00029     BEQL    2$               ;
                04 0002B     RET                     ; 0625
             00000000G  EF   9F 0002C 1$:   PUSHAB  CRD$GT_INTERNAL_ERROR          ; 0628
                01   DD 00032     PUSHL   #1               ;
                1A   DD 00034     PUSHL   #26              ;
             00000000G  FF       03  FB 00036     CALLS   #3, aCRD$PRINT          ;
             00000000G  EF   9F 0003D 2$:   PUSHAB  CRD$GT_SUMMARY_TITLE          ; 0635
                01   DD 00043     PUSHL   #1               ;
                1A   DD 00045     PUSHL   #26              ;
             00000000G  FF       03  FB 00047     CALLS   #3, aCRD$PRINT          ;
            0C  AE   D4 0004E     CLRL    I                ; 0638
                01 0C    AE  D1 00051 3$:   CMPL    I, #1            ; 0645
                0C   12 00055     BNEQ    4$               ;
                42 08    AE  E9 00057     BLBC    SOME_FAIL, 8$           ;
             00000000G  EF   9F 0005B     PUSHAB  CRD$GT_ALL_FAILURES          ;
                2F   11 00061     BRB     7$               ;
                02 0C    AE  D1 00063 4$:   CMPL    I, #2            ; 0648
                08   12 00067     BNEQ    5$               ;
             00000000G  EF   9F 00069     PUSHAB  CRD$GT_ALL_PASSES          ;
                21   11 0006F     BRB     7$               ;
                03 0C    AE  D1 00071 5$:   CMPL    I, #3            ; 0650
                0C   12 00075     BNEQ    6$               ;
                22 04    AE  E9 00077     BLBC    SOME_INC_PREP, 8$           ;
             00000000G  EF   9F 0007B     PUSHAB  CRD$GT_ALL_INC_HDWR_PREP         ;
                0F   11 00081     BRB     7$               ;
                04 0C    AE  D1 00083 6$:   CMPL    I, #4            ; 0651
                14   12 00087     BNEQ    8$               ;
                11       6E  E9 00089     BLBC    SOME_OTHER, 8$           ;
             00000000G  EF   9F 0008C     PUSHAB  CRD$GT_ALL_OTHERS           ;
                01   DD 00092 7$:   PUSHL   #1               ;
                1A   DD 00094     PUSHL   #26              ;
             00000000G  FF       03  FB 00096     CALLS   #3, aCRD$PRINT          ;
                57 00000000G    -   D0 0009D 8$:   MOVL    CRD$GA_FIRST_DIAGNOSTIC, DDB_PTR        ; 0654
                56   D4 000A4     CLRL    SCREEN_COUNTER           ; 0657
                02 0C    AE  D1 000A6     CMPL    I, #2            ; 0663
                14   12 000AA     BNEQ    9$               ;
```

```
                00000000G  EF  9F 000AC     PUSHAB  CRD$GT_INIT_ALL_PASSES          ; 0666
                    01  DD 000B2     PUSHL   #1                ;
                    1A  DD 000B4     PUSHL   #26               ;
                00000000G  FF    03  FB 000B6     CALLS   #3, @CRD$PRINT          ;
                    56      02  D0 000BD     MOVL    #2, SCREEN_COUNTER          ; 0667
            50      14   A7          01      05  EF 000C0 9$:   EXTZV   #5, #1, 20(DDB_PTR), R0        ; 0707
                    53      50  90 000C6     MOVB    R0, FINISHED              ;
            50      14   A7          01      02  EF 000C9     EXTZV   #2, #1, 20(DDB_PTR), R0        ; 0708
                    52      50  90 000CF     MOVB    R0, ESSENTIAL_DEVICE      ;
            50      14   A7          01      04  EF 000D2     EXTZV   #4, #1, 20(DDB_PTR), R0        ; 0709
                    51      50  90 000D8     MOVB    R0, BAD                   ;
            59      14   A7          01      01  EF 000DB     EXTZV   #1, #1, 20(DDB_PTR), R9        ; 0710
                    50      59  90 000E1     MOVB    R9, PREP_ERROR            ;
                    5A 08   A7  D0 000E4     MOVL    8(DDB_PTR), R10           ; 0713
                    29  13 000E8     BEQL    10$               ;
                    5B      53  9A 000EA     MOVZBL  FINISHED, R11             ; 0719
                    51      51  9A 000ED     MOVZBL  BAD, R1                 ;
                    51        5B      51  CB 000F0     BICL3   R1, R11, R1           ;
                    53      50  9A 000F4     MOVZBL  PREP_ERROR, R3            ; 0721
                    51      53  C8 000F7     BISL2   R3, R1                ;
                    59      51  92 000FA     MCOMB   R1, DDB_FAIL              ; 0719
            51          50          01      00  EF 000FD     EXTZV   #0, #1, PREP_ERROR, R1        ; 0726
                    53      51  90 00102     MOVB    R1, DDB_INC_PREP          ;
                    50      59  9A 00105     MOVZBL  DDB_FAIL, R0              ; 0731
                    5B      53  9A 00108     MOVZBL  DDB_INC_PREP, R11           ;
                    50      5B  C8 0010B     BISL2   R11, R0               ;
                    51      50  92 0010E     MCOMB   R0, DDB_PASS              ;
                    50  94 00111     CLRB    DDB_OTHER               ; 0736
                    5A  D5 00113 10$:   TSTL    R10                 ; 0740
                    0A  12 00115     BNEQ    11$               ;
                    59      52  90 00117     MOVB    ESSENTIAL_DEVICE, DDB_FAIL        ; 0743
                    53  94 0011A     CLRB    DDB_INC_PREP            ; 0744
                    51  94 0011C     CLRB    DDB_PASS                ; 0745
                    50      52  92 0011E     MCOMB   ESSENTIAL_DEVICE, DDB_OTHER         ; 0746
                    52      59  9A 00121 11$:   MOVZBL  DDB_FAIL, R2             ; 0754
        08  AE      52  C8 00124     BISL2   R2, SOME_FAIL             ;
                    52      53  9A 00128     MOVZBL  DDB_INC_PREP, R2            ; 0756
        04  AE      52  C8 0012B     BISL2   R2, SOME_INC_PREP             ;
                    52      50  9A 0012F     MOVZBL  DDB_OTHER, R2             ; 0758
                    6E      52  C8 00132     BISL2   R2, SOME_OTHER            ;
        0C  AE  D5 00135     TSTL    I                 ; 0762
                03  12 00138     BNEQ    12$               ;
            019E  31 0013A     BRW     31$               ;
                01 0C   AE  D1 0013D 12$:   CMPL    I, #1               ; 0763
                06  12 00141     BNEQ    13$               ;
                27      59  E8 00143     BLBS    DDB_FAIL, 16$             ;
            0192  31 00146     BRW     31$               ;
                02 0C   AE  D1 00149 13$:   CMPL    I, #2               ; 0764
                06  12 0014D     BNEQ    14$               ;
                1B      51  E8 0014F     BLBS    DDB_PASS, 16$             ;
            0186  31 00152     BRW     31$               ;
                03 0C   AE  D1 00155 14$:   CMPL    I, #3               ; 0765
                06  12 00159     BNEQ    15$               ;
                0F      53  E8 0015B     BLBS    DDB_INC_PREP, 16$           ;
            017A  31 0015E     BRW     31$               ;
```

```
            04 0C    AE  D1 00161 15$:   CMPL    I, #4              ; 0766
            06  12 00165    BNEQ    16$             ;
            03       50  E8 00167    BLBS    DDB_OTHER, 16$          ;
          016E  31 0016A    BRW     31$             ;
            58       5A  D0 0016D 16$:   MOVL    R10, PTABLE_PTR           ; 0775
            13  13 00170    BEQL    17$             ; 0783
            54 26    A8  9E 00172    MOVAB   38(R8), HDWR_NAME_PTR           ; 0786
            58  DD 00176    PUSHL   PTABLE_PTR          ; 0793
       00000000G  EF       01  FB 00178    CALLS   #1, CRD$GET_DEVICE_NAME         ;
            55       50  D0 0017F    MOVL    R0, DEV_NAME            ;
          00D2  31 00182    BRW     28$             ; 0783
            50 0C    A7  D0 00185 17$:   MOVL    12(DDB_PTR), R0           ; 0798
            55 0C    A0  D0 00189    MOVL    12(R0), DEV_NAME          ; 0803
            5B 0C    A0  D0 0018D    MOVL    12(R0), GENERIC_DEV_NAME          ; 0804
            5A  D4 00191    CLRL    J               ; 0848
            11  12 00193 18$:   BNEQ    19$             ; 0811
            59 00000000'  EF  9E 00195    MOVAB   P.AAC, TEMP_DEV_NAME          ; 0812
            54 00000000'  EF  9E 0019C    MOVAB   P.AAD, HDWR_NAME_PTR          ; 0813
          009B  31 001A3    BRW     27$             ; 0809
            01       5A  D1 001A6 19$:   CMPL    J, #1              ; 0815
            11  12 001A9    BNEQ    20$             ;
            59 00000000'  EF  9E 001AB    MOVAB   P.AAE, TEMP_DEV_NAME          ; 0816
            54 00000000'  EF  9E 001B2    MOVAB   P.AAF, HDWR_NAME_PTR          ; 0817
          0085  31 001B9    BRW     27$             ; 0809
            02       5A  D1 001BC 20$:   CMPL    J, #2              ; 0819
            10  12 001BF    BNEQ    21$             ;
            59 00000000'  EF  9E 001C1    MOVAB   P.AAG, TEMP_DEV_NAME          ; 0820
            54 00000000'  EF  9E 001C8    MOVAB   P.AAH, HDWR_NAME_PTR          ; 0821
            70  11 001CF    BRB     27$             ; 0809
            03       5A  D1 001D1 21$:   CMPL    J, #3              ; 0823
            10  12 001D4    BNEQ    22$             ;
            59 00000000'  EF  9E 001D6    MOVAB   P.AAI, TEMP_DEV_NAME          ; 0824
            54 00000000'  EF  9E 001DD    MOVAB   P.AAJ, HDWR_NAME_PTR          ; 0825
            5B  11 001E4    BRB     27$             ; 0809
            04       5A  D1 001E6 22$:   CMPL    J, #4              ; 0827
            10  12 001E9    BNEQ    23$             ;
            59 00000000'  EF  9E 001EB    MOVAB   P.AAK, TEMP_DEV_NAME          ; 0828
            54 00000000'  EF  9E 001F2    MOVAB   P.AAL, HDWR_NAME_PTR          ; 0829
            46  11 001F9    BRB     27$             ; 0809
            05       5A  D1 001FB 23$:   CMPL    J, #5              ; 0831
            10  12 001FE    BNEQ    24$             ;
            59 00000000'  EF  9E 00200    MOVAB   P.AAM, TEMP_DEV_NAME          ; 0832
            54 00000000'  EF  9E 00207    MOVAB   P.AAN, HDWR_NAME_PTR          ; 0833
            31  11 0020E    BRB     27$             ; 0809
            06       5A  D1 00210 24$:   CMPL    J, #6              ; 0835
            10  12 00213    BNEQ    25$             ;
            59 00000000'  EF  9E 00215    MOVAB   P.AAO, TEMP_DEV_NAME          ; 0836
            54 00000000'  EF  9E 0021C    MOVAB   P.AAP, HDWR_NAME_PTR          ; 0837
            1C  11 00223    BRB     27$             ; 0809
            07       5A  D1 00225 25$:   CMPL    J, #7              ; 0839
            10  12 00228    BNEQ    26$             ;
            59 00000000'  EF  9E 0022A    MOVAB   P.AAQ, TEMP_DEV_NAME          ; 0840
            54 00000000'  EF  9E 00231    MOVAB   P.AAR, HDWR_NAME_PTR          ; 0841
            07  11 00238    BRB     27$             ; 0809
            54 00000000G  EF  9E 0023A 26$:   MOVAB   CRD$GT_UNKNOWN, HDWR_NAME_PTR         ; 0844
```

```
                51       6B  9A 00241 27$:   MOVZBL  (GENERIC_DEV_NAME), R1          ; 0848
                50       69  9A 00244    MOVZBL  (TEMP_DEV_NAME), R0          ; 0849
           50          00   01   AB      51  2D 00247    CMPC5   R1, 1(GENERIC_DEV_NAME), #0, R0, -       ;
             01   A9      0024D    1(TEMP_DEV_NAME)          ;
                06  13 0024F    BEQL    28$             ;
         FF3C          5A         01      07  F1 00251    ACBL    #7, #1, J, 18$          ; 0806
                04 0C   AE  D1 00257 28$:   CMPL    I, #4              ; 0867
                1A  13 0025B    BEQL    30$             ;
                03       56  D1 0025D    CMPL    SCREEN_COUNTER, #3          ; 0870
                13  12 00260    BNEQ    29$             ;
             00000000G  EF  9F 00262    PUSHAB  CRD$GT_NEW_LINE          ; 0873
                01  DD 00268    PUSHL   #1              ;
                1A  DD 0026A    PUSHL   #26             ;
             00000000G  FF       03  FB 0026C    CALLS   #3, aCRD$PRINT          ;
                56  D4 00273    CLRL    SCREEN_COUNTER          ; 0874
                56  D6 00275 29$:   INCL    SCREEN_COUNTER          ; 0877
             00000000G  EF  9F 00277 30$:   PUSHAB  CRD$GT_TAB          ; 0880
                01  DD 0027D    PUSHL   #1              ;
                1A  DD 0027F    PUSHL   #26             ;
             00000000G  FF       03  FB 00281    CALLS   #3, aCRD$PRINT          ;
                54  DD 00288    PUSHL   HDWR_NAME_PTR           ; 0881
                01  DD 0028A    PUSHL   #1              ;
                1A  DD 0028C    PUSHL   #26             ;
             00000000G  FF       03  FB 0028E    CALLS   #3, aCRD$PRINT          ;
             00000000G  EF  9F 00295    PUSHAB  CRD$GT_SLASH          ; 0882
                01  DD 0029B    PUSHL   #1              ;
                1A  DD 0029D    PUSHL   #26             ;
             00000000G  FF       03  FB 0029F    CALLS   #3, aCRD$PRINT          ;
                55  DD 002A6    PUSHL   DEV_NAME            ; 0883
                01  DD 002A8    PUSHL   #1              ;
                1A  DD 002AA    PUSHL   #26             ;
             00000000G  FF       03  FB 002AC    CALLS   #3, aCRD$PRINT          ;
             00000000G  EF  9F 002B3    PUSHAB  CRD$GT_TAB          ; 0884
                01  DD 002B9    PUSHL   #1              ;
                1A  DD 002BB    PUSHL   #26             ;
             00000000G  FF       03  FB 002BD    CALLS   #3, aCRD$PRINT          ;
                04 0C    AE  D1 002C4    CMPL    I, #4              ; 0886
                11  12 002C8    BNEQ    31$             ;
             00000000G  EF  9F 002CA    PUSHAB  CRD$GT_DEV_UNAVAILABLE_2        ; 0888
                01  DD 002D0    PUSHL   #1              ;
                1A  DD 002D2    PUSHL   #26             ;
             00000000G  FF       03  FB 002D4    CALLS   #3, aCRD$PRINT          ;
                57       67  D0 002DB 31$:   MOVL    (DDB_PTR), DDB_PTR          ; 0898
             00000000G  EF       57  D1 002DE    CMPL    DDB_PTR, CRD$GA_FIRST_DIAGNOSTIC        ; 0902
                03  13 002E5    BEQL    32$             ;
              FDD6  31 002E7    BRW     9$              ;
                17       6E  E8 002EA 32$:   BLBS    SOME_OTHER, 33$          ; 0905
                04 0C    AE  D1 002ED    CMPL    I, #4              ; 0907
                11  12 002F1    BNEQ    33$             ;
             00000000G  EF  9F 002F3    PUSHAB  CRD$GT_NEW_LINE          ; 0910
                01  DD 002F9    PUSHL   #1              ;
                1A  DD 002FB    PUSHL   #26             ;
             00000000G  FF       03  FB 002FD    CALLS   #3, aCRD$PRINT          ;
         FD46     0C   AE         01      04  F1 00304 33$:   ACBL    #4, #1, I, 3$           ; 0638
               04 0030B    RET                 ; 0914
```

```
; Routine Size:  780 bytes,    Routine Base:  CODE + 036E


;   0915  1
```

```
;    0916  1 %SBTTL 'CRD$Screen_Pause Routine'
;    0917  1
;    0918  1 GLOBAL ROUTINE CRD$Screen_Pause (SCREEN_Operation) : NOVALUE =
;    0919  1
;    0920  1 !++
;    0921  1 ! Functional Description:
;    0922  1 !
;    0923  1 ! Handles output of CRD messages to soft console devices
;    0924  1 !
;    0925  1 ! Formal Input Parameters:
;    0926  1 !
;    0927  1 ! SCREEN_Operation: The soft console support operation that needs to be performed. Some
;    0928  1 !  operations apply only to VT52 or VT100 and will simply exit routine if terminal
;    0929  1 !  is hardcopy. Examples of this include Press Return for More prompt.
;    0930  1 !
;    0931  1 ! Formal Output Parameters:
;    0932  1 !
;    0933  1 ! None
;    0934  1 !
;    0935  1 ! Side Effects:
;    0936  1 !
;    0937  1 ! None
;    0938  1 !
;    0939  1 ! Completion Codes:
;    0940  1 !
;    0941  1 ! None
;    0942  1 !--
;    0943  2 BEGIN     ! Routine CRD$Screen_Pause
;    0944  2
;    0945  2  REGISTER
;    0946  2   Prompt : BYTE,
;    0947  2   Oper : BYTE;
;    0948  2  LOCAL
;    0949  2   Type_Clear_Screen,
;    0950  2   Prompt_Msg,
;    0951  2   Valid_Status,
;    0952  2   Local_OperInput_Buffer : $MEN_OperInput_Buffer_ATTR,
;    0953  2   Console_Char : $CRD_Termchar_ATTR;
;    0954  2
;    0955  2  BIND
;    0956  2   T_Prompt_Defau t = $ASCIC ('NULLRESPONSE') : VECTOR [,BYTE];
;    0957  2
;    0958  2
;    0959  2  !+
;    0960  2  ! BEFORE DOING ANYTHING, CHECK THAT THE CONSOLE IS SOFT (i.e. VT100 or VT52)
;    0961  2  ! AND NOT HARD (i.e. LA34 etc.)
;    0962  2  !-
;    0963  3  IF ($DS_GETTERM (TERMCHAR = Console_Char))
;    0964  3      ! If SS$_NORMAL status on getting
;    0965  3      ! terminal characteristics
;    0966  2  THEN
;    0967  3      BEGIN
;    0968  3   !+
;    0969  3   ! If console is not a CRD supported soft console, then just RETURN.
;    0970  3   ! Otherwise, continue normally.
```

```
;    0971  3     !-
;    0972  3         SELECTONE  (.Console_Char [Termchar$B_Type])  OF
;    0973  3     SET
;    0974  3        [TT$_VT52]:         TYPE_CLEAR_SCREEN = $ASCIC ('HJ');
;    0975  3        [TT$_VT100]:         TYPE_CLEAR_SCREEN = $ASCIC ('[H[J');
;    0976  3 !***      [TT$_VS125]:          TYPE_CLEAR_SCREEN = $ASCIC ('HJ');
;    0977  3        [TT$_VT200_Series]: TYPE_CLEAR_SCREEN = $ASCIC ('[2J');
;    0978  3        [OTHERWISE]:         RETURN;
;    0979  3     !+
;    0980  3     ! Record which type of soft console it is so that the
;    0981  3     ! correct escape sequence is sent when clearing the screen
;    0982  3     !-
;    0983  3     TES;
;    0984  3        END
;    0985  2    ELSE
;    0986  2         $CRD_Print (CRD$GT_Internal_Error);
;    0987  2
;    0988  2
;    0989  2
;    0990  2    !+
;    0991  2    ! Now that we know its a soft console, decide which operation
;    0992  2    ! to perform
;    0993  2    !-
;    0994  2    SELECTONE .SCREEN_Operation OF
;    0995  2     SET
;    0996  2     [SCREEN$K_Press_Return, SCREEN$K_Press_Return_MM,
;    0997  2     SCREEN$K_Press_Return_TM, SCREEN$K_TM_Msg]:
;    0998  3     BEGIN    ! Begin SCREEN$K_Press_Return_xx
;    0999  3
;    1000  3     !+
;    1001  3     ! Pause and prompt operator to press RETURN to continue
;    1002  3     !-
;    1003  3
;    1004  3     Valid_Status = False;  ! Assume the worst
;    1005  3
;    1006  3     Prompt = .DSA$V_Prompt;
;    1007  3     Oper   = .DSA$V_Oper;
;    1008  3
;    1009  3     CRD$GB_User_Prompt_Okay = True; ! Allow User_Prompt typecode to pass through CRD untouched.
;    1010  3     DSA$V_Prompt = On;  ! Turn on PROMPTing to allow prompts from the operator
;    1011  3     DSA$V_Oper   = On;  ! Turn on OPER to allow prompts from the operator
;    1012  3
;    1013  3     SELECTONE .SCREEN_Operation OF
;    1014  3      SET
;    1015  3      [SCREEN$K_Press_Return]: Prompt_Msg = CRD$GT_Press_Return;
;    1016  3
;    1017  3      [SCREEN$K_Press_Return_MM]: Prompt_Msg = CRD$GT_Press_Return_MM;
;    1018  3
;    1019  3      [SCREEN$K_Press_Return_TM]: Prompt_Msg = MEN$GT_Press_Return_TM;
;    1020  3
;    1021  3      [SCREEN$K_TM_Msg]: Prompt_Msg = MEN$GT_TM_Msg;
;    1022  3      TES;
;    1023  3
;    1024  3     DO
;    1025  4        BEGIN
```

```
;   1026  4
;   1027  4       $CRD_Print (CRD$GT_New_Line); ! Get prompt at start of the line
;   1028  4
; P 1029  5       IF ($MEN_ASKSTR (Msg_Adr = .Prompt_Msg,
; P 1030  5             Buf_Adr = Local_OperInput_Buffer,
;   1031  5             Def_Adr = T_Prompt_Default))
;   1032  4       THEN
;   1033  4       !+
;   1034  4       ! Valid response to prompt is either <CR/LF> or ^C followed by the 'Resume' choice
;   1035  4       ! from the ^C Menu. In both cases the default is deposited in the operator buffer.
;   1036  4       !-
;   1037  4
;   1038  7       IF (NOT (Valid_Status = (CH$EQL (.Local_OperInput_Buffer [0], Local_OperInput_Buffer [1],
;   1039  5                 .T_Prompt_Default [0],    T_Prompt_Default [1]))))
;   1040  4       THEN
;   1041  4        $CRD_Message (New_Line, MEN$GT_InvOperInput);
;   1042  4       END
;   1043  3    UNTIL (.Valid_Status);  ! Repeat this till we get a valid answer
;   1044  3
;   1045  3    $CRD_Print (.Type_Clear_Screen);
;   1046  3        ! Start at top of screen after operator presses return
;   1047  3
;   1048  3    CRD$GB_User_Prompt_Okay = False; ! Disallow User_Prompt typecode to pass through CRD untouched.
;   1049  3    DSA$V_Prompt = .Prompt;  ! Restore PROMPT flag
;   1050  3    DSA$V_Oper   = .Oper;  ! Restore OPER flag
;   1051  3
;   1052  2    END;     ! End SCREEN$K_Press_Return
;   1053  2
;   1054  2
;   1055  2   [SCREEN$K_Clear_Screen]:
;   1056  3    BEGIN    ! Begin SCREEN$K_Clear_Screen
;   1057  4    $CRD_Print (.Type_Clear_Screen)
;   1058  4        ! Print an escape sequence
;   1059  2    END;     ! End SCREEN$K_Clear_Screen
;   1060  2
;   1061  2   [SCREEN$K_Count_Line]:
;   1062  3    BEGIN    ! Begin SCREEN$K_Count_Line
;   1063  3    RETURN    ! *** TEMPORARY
;   1064  2    END;     ! End SCREEN$K_Count_Line
;   1065  2
;   1066  2   [OTHERWISE]:
;   1067  3    BEGIN
;   1068  4    $CRD_Print (CRD$GT_Internal_Error)
;   1069  2    END;
;   1070  2
;   1071  2   TES;
;   1072  1 END;     ! Routine CRD$Screen_Pause


          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

45  53  4E  4F  50  53  45  52  4C  4C  55  4E  0C  000AF P.AAS:  .ASCII  <12>\NULLRESPONSE\         ;
    4A  1B  48  1B  04  000BC P.AAT:  .ASCII  <4><27>\H\<27>\J\          ;
   4A  5B  1B  48  5B  1B  06  000C1 P.AAU:  .ASCII  <6><27>\[H\<27>\[J\          ;
```

```
     4A  32  5B  1B  04  000C8 P.AAV:  .ASCII  <4><27>\[2J\          ;

          T_PROMPT_DEFAULT=   P.AAS
           .EXTRN  CRD$GB_USER_PROMPT_OKAY
           .EXTRN  CRD$GT_PRESS_RETURN
           .EXTRN  CRD$GT_PRESS_RETURN_MM
           .EXTRN  MEN$GT_PRESS_RETURN_TM
           .EXTRN  MEN$GT_TM_MSG, DS$BREAK
           .EXTRN  DS$ASKSTR, MEN$GT_INVOPERINPUT
           .EXTRN  CRD$GT_NEW_LINE_MESSAGE

           .PSECT  CODE,NOWRT,  SHR,  PIC,2

             03FC 00000     .ENTRY  CRD$SCREEN_PAUSE, Save R2,R3,R4,R5,R6,R7,-   ; 0918
            R8,R9            ;
           5E     FEF8   CE  9E 00002     MOVAB   -264(SP), SP          ;
           5E   DD 00007     PUSHL   SP                ; 0963
      00000000G  9F       01  FB 00009     CALLS   #1, a#DS$GETTERM          ;
           32       50  E9 00010     BLBC    R0, 4$            ;
           50 01    AE  9A 00013     MOVZBL  CONSOLE_CHAR+1, R0          ; 0972
      40   8F       50  91 00017     CMPB    R0, #64           ; 0974
           09  12 0001B     BNEQ    1$              ;
           59 00000000'  EF  9E 0001D     MOVAB   P.AAT, TYPE_CLEAR_SCREEN        ;
           30  11 00024     BRB     5$              ;
      60   8F       50  91 00026 1$:   CMPB    R0, #96           ; 0975
           09  12 0002A     BNEQ    2$              ;
           59 00000000'  EF  9E 0002C     MOVAB   P.AAU, TYPE_CLEAR_SCREEN        ;
           21  11 00033     BRB     5$              ;
        6E   8F       50  91 00035 2$:   CMPB    R0, #110          ; 0977
           01  13 00039     BEQL    3$              ;
            04 0003B     RET                ;
           59 00000000'  EF  9E 0003C 3$:   MOVAB   P.AAV, TYPE_CLEAR_SCREEN        ;
           11  11 00043     BRB     5$              ;
      00000000G  EF  9F 00045 4$:   PUSHAB  CRD$GT_INTERNAL_ERROR         ; 0986
           01  DD 0004B     PUSHL   #1              ;
           1A  DD 0004D     PUSHL   #26             ;
      00000000G  FF       03  FB 0004F     CALLS   #3, aCRD$PRINT          ;
           52 04    AC  D0 00056 5$:   MOVL    SCREEN_OPERATION, R2         ; 0994
           10  13 0005A     BEQL    7$              ; 0996
           03       52  D1 0005C     CMPL    R2, #3          ;
           03  18 0005F     BGEQ    6$              ;
            0114  31 00061     BRW     15$             ;
           05       52  D1 00064 6$:   CMPL    R2, #5          ;
           03  15 00067     BLEQ    7$              ;
            010C  31 00069     BRW     15$             ;
           58  D4 0006C 7$:   CLRL    VALID_STATUS          ; 1004
     50 0000FE01   9F         01      05  EF 0006E     EXTZV   #5, #1, a#^X0000FE01, R0        ; 1006
           54       50  90 00077     MOVB    R0, PROMPT              ;
     50 0000FE01   9F         01      04  EF 0007A     EXTZV   #4, #1, a#^X0000FE01, R0        ; 1007
           55       50  90 00083     MOVB    R0, OPER                ;
      00000000G  EF       01  90 00086     MOVB    #1, CRD$GB_USER_PROMPT_OKAY         ; 1009
      0000FE01   9F       30  88 0008D     BISB2   #48, a#^X0000FE01           ; 1011
           52  D5 00094     TSTL    R2              ; 1015
           09  12 00096     BNEQ    8$              ;
           57 00000000G  EF  9E 00098     MOVAB   CRD$GT_PRESS_RETURN, PROMPT_MSG         ;
```

```
                28  11 0009F     BRB     11$                ;
                03       52  D1 000A1 8$:    CMPL    R2, #3           ; 1017
                09  12 000A4     BNEQ    9$                 ;
                57 00000000G  EF  9E 000A6    MOVAB   CRD$GT_PRESS_RETURN_MM, PROMPT_MSG       ;
                1A  11 000AD     BRB     11$                ;
                04       52  D1 000AF 9$:    CMPL    R2, #4           ; 1019
                09  12 000B2     BNEQ    10$                ;
                57 00000000G  EF  9E 000B4    MOVAB   MEN$GT_PRESS_RETURN_TM, PROMPT_MSG       ;
                0C  11 000BB     BRB     11$                ;
                05       52  D1 000BD 10$:    CMPL    R2, #5           ; 1021
                07  12 000C0     BNEQ    11$                ;
                57 00000000G  EF  9E 000C2    MOVAB   MEN$GT_TM_MSG, PROMPT_MSG        ;
             00000000G  EF  9F 000C9 11$:    PUSHAB  CRD$GT_NEW_LINE          ; 1027
                01  DD 000CF     PUSHL   #1                 ;
                1A  DD 000D1     PUSHL   #26                ;
             00000000G  FF       03  FB 000D3    CALLS   #3, @CRD$PRINT           ;
             00000000G  9F       00  FB 000DA    CALLS   #0, @#DS$BREAK           ; 1031
                7E  D4 000E1     CLRL    -(SP)              ;
             00000000'  EF  9F 000E3    PUSHAB  T_PROMPT_DEFAULT         ;
                7E  D4 000E9     CLRL    -(SP)              ;
                7E 48   8F  9A 000EB    MOVZBL  #72, -(SP)         ;
             18   AE  9F 000EF     PUSHAB  LOCAL_OPERINPUT_BUFFER       ;
                57  DD 000F2     PUSHL   PROMPT_MSG         ;
             00000000G  9F       06  FB 000F4    CALLS   #6, @#DS$ASKSTR          ;
                52       50  D0 000FB     MOVL    R0, STATUS         ;
             00000000G  9F       00  FB 000FE    CALLS   #0, @#DS$BREAK           ;
                44       52  E9 00105     BLBC    STATUS, 13$        ;
                51 08    AE  9A 00108     MOVZBL  LOCAL_OPERINPUT_BUFFER, R1        ; 1038
                50 00000000'  EF  9A 0010C    MOVZBL  T_PROMPT_DEFAULT, R0         ; 1039
                56  D4 00113     CLRL    R6                 ;
          50          00   09    AE      51  2D 00115    CMPC5   R1, LOCAL_OPERINPUT_BUFFER+1, #0, R0, -     ;
             00000000'  EF      0011B    T_PROMPT_DEFAULT+1         ;
                02  12 00120     BNEQ    12$                ;
                56  D6 00122     INCL    R6                 ;
                58       56  D0 00124 12$:    MOVL    R6, VALID_STATUS         ; 1038
                22       56  E8 00127     BLBS    R6, 13$            ;
             00000000G  EF  9F 0012A     PUSHAB  CRD$GT_NEW_LINE_MESSAGE          ; 1041
                01  DD 00130     PUSHL   #1                 ;
                1A  DD 00132     PUSHL   #26                ;
             00000000G  FF       03  FB 00134    CALLS   #3, @CRD$PRINT           ;
             00000000G  EF  9F 0013B     PUSHAB  MEN$GT_INVOPERINPUT          ;
                01  DD 00141     PUSHL   #1                 ;
                1A  DD 00143     PUSHL   #26                ;
             00000000G  FF       03  FB 00145    CALLS   #3, @CRD$PRINT           ;
                03       58  E8 0014C 13$:    BLBS    VALID_STATUS, 14$         ; 1043
              FF77  31 0014F     BRW     11$                ;
                59  DD 00152 14$:    PUSHL   TYPE CLEAR_SCREEN          ; 1045
                01  DD 00154     PUSHL   #1                 ;
                1A  DD 00156     PUSHL   #26                ;
             00000000G  FF       03  FB 00158    CALLS   #3, @CRD$PRINT           ;
             00000000G  EF  94 0015F     CLRB    CRD$GB_USER_PROMPT_OKAY          ; 1048
0000FE01   9F          01        05      54  F0 00165    INSV    PROMPT, #5, #1, @#^X0000FE01         ; 1049
0000FE01   9F          01        04      55  F0 0016E    INSV    OPER, #4, #1, @#^X0000FE01         ; 1050
              04 00177     RET                 ; 0994
                01       52  D1 00178 15$:    CMPL    R2, #1           ; 1055
```

```
            04  12 0017B     BNEQ    16$             ;
            59  DD 0017D     PUSHL   TYPE_CLEAR_SCREEN          ; 1057
            0B  11 0017F     BRB     17$             ;
            02       52  D1 00181 16$:   CMPL    R2, #2           ; 1061
            11  13 00184     BEQL    18$             ;
       00000000G  EF  9F 00186     PUSHAB  CRD$GT_INTERNAL_ERROR          ; 1068
            01  DD 0018C 17$:   PUSHL   #1              ;
            1A  DD 0018E     PUSHL   #26             ;
       00000000G  FF       03  FB 00190     CALLS   #3, @CRD$PRINT          ;
        04 00197 18$:   RET                 ; 1072

; Routine Size:  408 bytes,    Routine Base:  CODE + 067A


;   1073  1 END     ! End of CRDSTOP module
;   1074  0 ELUDOM




;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          205  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK          104  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE         2066  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$ASCIC     Macro    Lib02   452     812     813     816     817     820     821     824     825     828
              829      832     833     836     837     840     841     956     974     975
              977
$CRD_LITERALS   Macro    Lib03   154
$CRD_MESSAGE    Macro    Lib03  1041
$CRD_PRINT    Unbound          1041
    Macro    Lib03   225     227     238     243     248     254     255     260     265     270
              284     289     294     368     375     377     381     382     398     452
              628     635     645     648     650     651     666     873     880     881
              882     883     884     888     910     986    1027    1041    1045    1057
             1068
$CRD_TERMCHAR_ATTR  Macro    Lib03   592     953
$CRD_TERMCHAR_FLD  Fieldset Lib03   592     953
$DS_ASKSTR    KeyWMacr Lib02  1031
$DS_BREAK    Macro    Lib02  1031
$DS_DSADEF    Macro    Lib02   155
$DS_GETTERM    KeyWMacr Lib02   610     963
$DS_HPO_DECL    Macro    Lib02   600
$DS_HPO_F    Fieldset Lib02   600
$DS_TYPEDEF    Macro    Lib02   225     227     238     243     248     254     255     260     265     270
              284     289     294     368     375     377     381     382     398     454
              628     635     645     648     650     651     666     873     880     881
              882     883     884     888     910     986    1027    1041    1045    1057
             1068
$EQULST     Macro    Lib04   154     156     225     227     238     243     248     254     255     260
              265     270     284     289     294     368     375     377     381     382
              398     454     628     635     645     648     650     651     666     873
              880     881     882     883     884     888     910     986    1027    1041
             1045    1057    1068
$ER     Comptime   172
$MEN_ASKSTR    KeyWMacr Lib03  1029
$MEN_OPERINPUT_BUFFER_ATTR Macro    Lib03   952
$MEN_SUPBLK_ATTR  Macro    Lib03   599
$MEN_SUPBLK_FLD   Fieldset Lib03   599
$MODULE     Bind         172
$SCREEN_LITERALS  Macro    Lib03   156
AUTO$K_EXIT_AUTOSIZER_ERROR Literal    154   371
AUTO$K_EXIT_CONTROL_C  Literal    154   246     554
AUTO$K_EXIT_DUMMY_2  Literal    154
AUTO$K_EXIT_DUMMY_3  Literal    154
AUTO$K_EXIT_ERRORS_COMPLETE Literal    154   263     388
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal    154   297     390
AUTO$K_EXIT_INTERNAL_ERROR Literal    154   252
AUTO$K_EXIT_INV_BASE_SYSTEM Literal    154   379
AUTO$K_EXIT_LAST_REASON  Literal    154   154
AUTO$K_EXIT_NOERRORS_COMPLETE Literal    154   258     387     401     461     479
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal    154   287     389     402     462     480
AUTO$K_EXIT_NOERRORS_PREP Literal    154   292     391     403     463     481
BAD     Register   699   709=    719.
BOOLEAN     Macro    Lib03   726
CHAR_PTR    Local      303   313=    343a=   345=    345.    355a=   357=    357.    366.
CONSOLE_CHAR    Local      592   610a    619.    621.    622.
    Local      953   963a    972.
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------- -------- ----- ---------------
CRD$AL_CRD_DISPATCH_VECTORS External Lib03   209a
CRD$AUTO_SUMMARY  GlobRout   561 Lib03e    147f    395c
CRD$CLEAR_CTRL_C_FLAG  ExtRout  Lib03   231c    552c
CRD$CONTROL_C_HANDLER  GlobRout   527 Lib03e    141f
CRD$DISABLE_CTRL_C  ExtRout  Lib03   230c    551c
CRD$DISPOSE   ExtRout  Lib03   434c
CRD$DSR$CHECK_AUTOTEST_OFF_SET External Lib03   216.
CRD$DSR$CHECK_MENUTEST_OFF_SET External Lib03   215.
CRD$DS_INTERFACE  ExtRout  Lib03   521c
CRD$EXIT_DS   External Lib03   487.
CRD$GA_BEGIN   External Lib03   523ac
CRD$GA_CRD_MESSAGE_BUFFER External Lib03   313a    365=    366a    368a
CRD$GA_FIRST_DIAGNOSTIC  External Lib03   300m    315.    363.    418.    440.
    Map         603   654.    902.
CRD$GB_AUTO_TO_MENU_OFF  Global     176   176=    223.    228=    518=
CRD$GB_USER_PROMPT_OKAY  External Lib03  1009=   1048=
CRD$GET_DEVICE_NAME  ExtRout  Lib03   793c
CRD$GL_CRD_DEBUG_VECTOR  Global     179 Lib03e    503.
CRD$GT_ALL_FAILURES  External Lib03   645a
CRD$GT_ALL_INC_HDWR_PREP External Lib03   650a
CRD$GT_ALL_OTHERS  External Lib03   651a
CRD$GT_ALL_PASSES  External Lib03   648a
CRD$GT_CONTROL_C_ABORT  External Lib03   248a
CRD$GT_CRD_BAD_DEVICE  External Lib03   368a
CRD$GT_CRD_DEV_UNTESTED  External Lib03   294a
CRD$GT_CRD_END_MESSAGE  External Lib03   398a
CRD$GT_CRD_ERRORS_COMPLETE External Lib03   265a
CRD$GT_CRD_EVSBA_ERROR  External Lib03   375a
CRD$GT_CRD_EVSBB_ERROR  External Lib03   377a
CRD$GT_CRD_FATAL_ERROR  External Lib03   382a
CRD$GT_CRD_INCOMPLETE  External Lib03   289a
CRD$GT_CRD_NO_ERRORS  External Lib03   260a
CRD$GT_DEV_UNAVAILABLE_2 External Lib03   888a
CRD$GT_EXIT_TO_CLI  External Lib03   227a
CRD$GT_EXIT_TO_CONSOLE  External Lib03   225a
CRD$GT_INIT_ALL_PASSES  External Lib03   666a
CRD$GT_INTERNAL_ERROR  External Lib03   254a    628a    986a   1068a
CRD$GT_INTERNAL_ERROR_ABORT External Lib03   255a
CRD$GT_INV_BASE_SYSTEM  External Lib03   381a
CRD$GT_NEW_LINE   External Lib03   873a    910a   1027a
CRD$GT_NEW_LINE_MESSAGE  External Lib03  1041a
CRD$GT_PRESS_RETURN  External Lib03  1015a
CRD$GT_PRESS_RETURN_MM  External Lib03  1017a
CRD$GT_SLASH   External Lib03   882a
CRD$GT_SPACE_AND_SPACE  External Lib03   338m    341.    342.    345.
CRD$GT_STAR_LINE_PREFIX_MESSAGE Unbound         1041
CRD$GT_STAR_LINE_SUFFIX_MESSAGE Unbound         1041
CRD$GT_SUMMARY_TITLE  External Lib03   635a
CRD$GT_TAB   External Lib03   880a    884a
CRD$GT_UNKNOWN   External Lib03   844a
CRD$GT_WHICH_CRD  External Lib03   248.    254.    255.    260.    265.    289.    294.    368.    375.    377.
             382.    398.
CRD$K_ACTION_EQL_DO_NOTHING Literal    154
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_ACTION_RANGE_ERROR Literal     154
CRD$K_ADVANCE_DIAGNOSTIC Literal     154
CRD$K_ADVANCE_GENERAL_COM Literal     154
CRD$K_ADVANCE_STATE  Literal     154
CRD$K_ALLOCATION_ERROR  Literal     154
CRD$K_ASSIGN_PTABLE_PTRS Literal     154
CRD$K_AUTOSIZER_ERROR  Literal     154
CRD$K_AUTOSIZER_SET_FLAGS Literal     154
CRD$K_BAD_ACTION_NUMBER  Literal     154
CRD$K_BAD_TYPECODE_ERROR Literal     154
CRD$K_BASE_SYSTEM_IDC  Literal     154
CRD$K_BASE_SYSTEM_LESI  Literal     154
CRD$K_BASE_SYSTEM_NULL  Literal     154
CRD$K_BASE_SYSTEM_UDA_0  Literal     154
CRD$K_BASE_SYSTEM_UDA_1  Literal     154
CRD$K_BASE_SYSTEM_UDA_2  Literal     154
CRD$K_BASE_SYSTEM_UDA_3  Literal     154
CRD$K_CRD_INITIALIZATION Literal     154   521
CRD$K_CREATE_HST  Literal     154
CRD$K_DATA_LENGTH_ERROR  Literal     154
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     154   322     798
CRD$K_DDB_BLINK   Literal     154
CRD$K_DDB_FLINK   Literal     154   360     432     898
CRD$K_DDB_LAST_ENTRY  Literal     154
CRD$K_DDB_MEMORY_SIZE  Literal     154
CRD$K_DDB_MENU_DEVICE_NUMB Literal     154
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     154
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     154
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     154
CRD$K_DDB_PTABLE_ENTRY  Literal     154   713     740     775
CRD$K_DDB_STATUS  Literal     154   319     427     707     708     709     710
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     154
CRD$K_DEVICE_NAME  Literal     154   326     803     804
CRD$K_DIAGNOSTIC_ERROR  Literal     154
CRD$K_DIAGNOSTIC_NAME  Literal     154
CRD$K_DONE_GENERAL_COMS  Literal     154
CRD$K_DO_NOTHING  Literal     154
CRD$K_DUMMY_10   Literal     154
CRD$K_DUMMY_11   Literal     154
CRD$K_DUMMY_12   Literal     154
CRD$K_DUMMY_13   Literal     154
CRD$K_DUMMY_3   Literal     154
CRD$K_DUMMY_4   Literal     154
CRD$K_DUMMY_5   Literal     154
CRD$K_DUMMY_6   Literal     154
CRD$K_DUMMY_7   Literal     154
CRD$K_DUMMY_8   Literal     154
CRD$K_DUMMY_9   Literal     154
CRD$K_EXIT_CRD   Literal     154
CRD$K_HOOK_POINT  Literal     154   521
CRD$K_HS_INTERNAL_ERROR  Literal     154
CRD$K_IDC_EVDLB   Literal     154
CRD$K_IDC_EVDLC   Literal     154
```

Symbol	Type	Defined	Referenced ...
CRD$K_IDC_EVDLD	Literal	154	
CRD$K_IDC_EVRAD_0	Literal	154	
CRD$K_IDC_EVRAD_1	Literal	154	
CRD$K_IDC_LAST_DIAGNOSTIC	Literal	154	
CRD$K_INTERNAL_ERROR	Literal	154	
CRD$K_LAST_ACTION_ROUTINE	Literal	154	
CRD$K_LAST_ENTRY	Literal	154	
CRD$K_LAST_FUNCTION	Literal	154	
CRD$K_LAST_HOOK_POINT	Literal	154	
CRD$K_LAST_INTERNAL_ERROR	Literal	154	
CRD$K_LAST_TYPE	Literal	154	
CRD$K_LESI_ENWCA	Literal	154	
CRD$K_LESI_EVRMB_0	Literal	154	
CRD$K_LESI_EVRMB_1	Literal	154	
CRD$K_LESI_LAST_DIAGNOSTIC	Literal	154	
CRD$K_LEVEL_4_PASSED	Literal	154	
CRD$K_LOAD_AUTOSIZER	Literal	154	
CRD$K_LOAD_ERROR	Literal	154	
CRD$K_LOAD_GENERAL_COM	Literal	154	
CRD$K_LOAD_LOAD_COM	Literal	154	
CRD$K_LOAD_SELECT_COM	Literal	154	
CRD$K_LOAD_SET_EVENT_COM	Literal	154	
CRD$K_LOAD_SET_FLAGS_COM	Literal	154	
CRD$K_LOAD_START_COM	Literal	154	
CRD$K_LOAD_SUP_FILE	Literal	154	
CRD$K_MM_INTERNAL_ERROR	Literal	154	
CRD$K_NEW_LINE	Literal	154	
CRD$K_NUMB_DISPATCH_VECTORS	Literal	Lib03	492
CRD$K_NUMB_UNUSED_VECTORS	Literal	Lib03	492
CRD$K_PREPARATION_ERROR	Literal	154	
CRD$K_PREPARATION_ERROR_TEXT	Literal	154	
CRD$K_RUNTIME	Literal	154	
CRD$K_SAME_LINE	Literal	154	
CRD$K_SET_EVENT_ARGS	Literal	154	
CRD$K_SET_FLAGS_ARGS	Literal	154	
CRD$K_SET_UP_DIAGNOSTIC	Literal	154	
CRD$K_START	Literal	154	
CRD$K_START_AUTOSIZER	Literal	154	
CRD$K_START_ERROR	Literal	154	
CRD$K_START_QUALIFIERS	Literal	154	
CRD$K_STAR_LINE	Literal	154	
CRD$K_STATE_ADVANCE_DIAGNOSTIC	Literal	154	
CRD$K_STATE_ADVANCE_GENERAL_COM	Literal	154	
CRD$K_STATE_ASSIGN_PTABLE_PTRS	Literal	154	
CRD$K_STATE_AUTOSIZER_ERROR	Literal	154	
CRD$K_STATE_AUTOSIZER_RUNNING	Literal	154	
CRD$K_STATE_AUTOSIZER_SET_FLAGS	Literal	154	
CRD$K_STATE_DIAGNOSTIC_ERROR	Literal	154	
CRD$K_STATE_DIAGNOSTIC_RUNNING	Literal	154	
CRD$K_STATE_DONE_DIAGNOSTICS	Literal	154	
CRD$K_STATE_DONE_GENERAL_COMS	Literal	154	
CRD$K_STATE_DUMMY_1	Literal	154	
CRD$K_STATE_DUMMY_10	Literal	154	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_DUMMY_12  Literal    154
CRD$K_STATE_DUMMY_13  Literal    154
CRD$K_STATE_DUMMY_2  Literal    154
CRD$K_STATE_DUMMY_3  Literal    154
CRD$K_STATE_DUMMY_4  Literal    154
CRD$K_STATE_DUMMY_6  Literal    154
CRD$K_STATE_DUMMY_7  Literal    154
CRD$K_STATE_DUMMY_8  Literal    154
CRD$K_STATE_EXIT_CRD  Literal    154
CRD$K_STATE_INTERNAL_ERROR Literal    154
CRD$K_STATE_LAST_STATE  Literal    154
CRD$K_STATE_LEVEL_4_PASSED Literal    154
CRD$K_STATE_LOAD_AUTOSIZER Literal    154
CRD$K_STATE_LOAD_ERROR  Literal    154
CRD$K_STATE_LOAD_GENERAL_COM Literal    154
CRD$K_STATE_LOAD_LOAD_COM Literal    154
CRD$K_STATE_LOAD_SELECT_COM Literal    154
CRD$K_STATE_LOAD_SET_EVENT_COM Literal    154
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal    154
CRD$K_STATE_LOAD_START_COM Literal    154
CRD$K_STATE_PREPARATION_ERROR Literal    154
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal    154
CRD$K_STATE_START  Literal    154
CRD$K_STATE_START_AUTOSIZER Literal    154
CRD$K_STATE_START_ERROR  Literal    154
CRD$K_TERMCHAR_SIZE  Literal  Lib03   592      953
CRD$K_TEST_START_TEXT  Literal    154
CRD$K_TOTAL_NUMB_VECTORS Literal  Lib03   179      492      501
CRD$K_TQ_INTERNAL_ERROR  Literal    154
CRD$K_TYPECODE   Literal    154
CRD$K_UDA_0_EVDLB  Literal    154
CRD$K_UDA_0_EVDLC  Literal    154
CRD$K_UDA_0_EVDLD  Literal    154
CRD$K_UDA_0_EVRLA_0  Literal    154
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal    154
CRD$K_UDA_1_EVDLB  Literal    154
CRD$K_UDA_1_EVDLC  Literal    154
CRD$K_UDA_1_EVDLD  Literal    154
CRD$K_UDA_1_EVRLA_0  Literal    154
CRD$K_UDA_1_EVRLA_1  Literal    154
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal    154
CRD$K_UDA_2_EVDLB  Literal    154
CRD$K_UDA_2_EVDLC  Literal    154
CRD$K_UDA_2_EVDLD  Literal    154
CRD$K_UDA_2_EVRLA_0  Literal    154
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal    154
CRD$K_UDA_3_EVDLB  Literal    154
CRD$K_UDA_3_EVDLC  Literal    154
CRD$K_UDA_3_EVDLD  Literal    154
CRD$K_UDA_3_EVRLA_0  Literal    154
CRD$K_UDA_3_EVRLA_1  Literal    154
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal    154
CRD$PRINT   External Lib03   225ac   227ac   238ac   243ac   248ac   254ac   255ac   260ac   265ac   270ac
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
             284ac   289ac   294ac   368ac   375ac   377ac   381ac   382ac   398ac   454ac
             628ac   635ac   645ac   648ac   650ac   651ac   666ac   873ac   880ac   881ac
             882ac   883ac   884ac   888ac   910ac   986ac  1027ac  1041ac  1045ac  1057ac
            1068ac
CRD$PRINT_DDB   ExtRout  Lib03   430c
CRD$SCREEN_PAUSE  GlobRout   918 Lib03e    150f    467c
CRD$STOP   GlobRout   184 Lib03e    144f    554c
CRD_ASSERT   Macro    Lib03   490
CRD_DISPATCH_VECTORS  Bind        209   503=
DDB$B_STATUS_DDB_SIZE  Macro    Lib03   427
DDB$V_STATUS_DEVICE_BAD  Macro    Lib03   319     709
DDB$V_STATUS_DEV_TEST_COMPLETE Macro    Lib03   707
DDB$V_STATUS_ESSENTIAL_DEVICE Macro    Lib03   708
DDB$V_STATUS_PREPARATION_ERROR Macro    Lib03   710
DDB_FAIL   Register   702   719=    731.    743=    754.    763.
DDB_INC_PREP   Register   705   726=    731.    744=    756.    765.
DDB_OTHER   Register   703   736=    746=    758.    766.
DDB_PASS   Register   704   731=    745=    764.
DDB_PTR    Register   307   315=    319a.   322a.   360=    360a.   363.
    Register   411   418=    427a.   430.    432a.   434.    437=    440.
    Register   598   654=    707a.   708a.   709a.   710a.   713a.   740a.   775a.   798a.   898=
            898a.   902.
DEVICE_DATA_BLOCK_STRUCTURE Structur Lib03   300     307     411     598     603
DEVICE_NAME   Bind        326   353.    354.    357.
DEV_NAME   Register   596   793=    803=    883.
DIAG_SUPPORT_VECTOR  Bind        798   803.    804.
DIAG_VECTOR_PTR   Bind        322   326.
DS$ASKSTR   ExtRout   1031  1031c
DS$BREAK   ExtRout   1031  1031c   1031e
DS$GETTERM   ExtRout    610   610c    963e    963c
DS$K_PRINTB   Literal    225
    Literal    227
    Literal    238
    Literal    243
    Literal    248
    Literal    254
    Literal    255
    Literal    260
    Literal    265
    Literal    270
    Literal    284
    Literal    289
    Literal    294
    Literal    368
    Literal    375
    Literal    377
    Literal    381
    Literal    382
    Literal    398
    Literal    454
    Literal    628
    Literal    635
    Literal    645
```

Symbol	Type	Defined	Referenced ...
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_PRINTF	Literal	225	225
	Literal	227	227
	Literal	238	238
	Literal	243	243
	Literal	248	248
	Literal	254	254
	Literal	255	255
	Literal	260	260
	Literal	265	265
	Literal	270	270
	Literal	284	284
	Literal	289	289
	Literal	294	294
	Literal	368	368
	Literal	375	375
	Literal	377	377
	Literal	381	381
	Literal	382	382
	Literal	398	398
	Literal	454	454
	Literal	628	628
	Literal	635	635
	Literal	645	645
	Literal	648	648
	Literal	650	650
	Literal	651	651
	Literal	666	666
	Literal	873	873
	Literal	880	880
	Literal	881	881
	Literal	882	882
	Literal	883	883
	Literal	884	884
	Literal	888	888

Symbol	Type	Defined	Referenced ...
	Literal	910	910
	Literal	986	986
	Literal	1027	1027
	Literal	1041	1041
	Literal	1041	1041
	Literal	1045	1045
	Literal	1057	1057
	Literal	1068	1068
DS$K_PRINTI	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_PRINTX	Literal	225	
	Literal	227	
	Literal	238	

Symbol	Type	Defined	Referenced ...
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_ABORT_PROGRAM	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	

Symbol	Type	Defined	Referenced ...
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_ABORT_TEST	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	

Symbol	Type	Defined	Referenced ...
	Literal 651		
	Literal 666		
	Literal 873		
	Literal 880		
	Literal 881		
	Literal 882		
	Literal 883		
	Literal 884		
	Literal 888		
	Literal 910		
	Literal 986		
	Literal 1027		
	Literal 1041		
	Literal 1041		
	Literal 1045		
	Literal 1057		
	Literal 1068		
DS$K_TYPE_COMMAND_ERR	Literal 225		
	Literal 227		
	Literal 238		
	Literal 243		
	Literal 248		
	Literal 254		
	Literal 255		
	Literal 260		
	Literal 265		
	Literal 270		
	Literal 284		
	Literal 289		
	Literal 294		
	Literal 368		
	Literal 375		
	Literal 377		
	Literal 381		
	Literal 382		
	Literal 398		
	Literal 454		
	Literal 628		
	Literal 635		
	Literal 645		
	Literal 648		
	Literal 650		
	Literal 651		
	Literal 666		
	Literal 873		
	Literal 880		
	Literal 881		
	Literal 882		
	Literal 883		
	Literal 884		
	Literal 888		
	Literal 910		
	Literal 986		

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------

    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_COMMAND_OUT  Literal    225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_CRD_AUTOTEST  Literal    225   225
    Literal     227   227
    Literal     238   238
    Literal     243   243
    Literal     248   248
```

Symbol	Type	Defined	Referenced ...
	Literal	254	254
	Literal	255	255
	Literal	260	260
	Literal	265	265
	Literal	270	270
	Literal	284	284
	Literal	289	289
	Literal	294	294
	Literal	368	368
	Literal	375	375
	Literal	377	377
	Literal	381	381
	Literal	382	382
	Literal	398	398
	Literal	454	454
	Literal	628	628
	Literal	635	635
	Literal	645	645
	Literal	648	648
	Literal	650	650
	Literal	651	651
	Literal	666	666
	Literal	873	873
	Literal	880	880
	Literal	881	881
	Literal	882	882
	Literal	883	883
	Literal	884	884
	Literal	888	888
	Literal	910	910
	Literal	986	986
	Literal	1027	1027
	Literal	1041	1041
	Literal	1041	1041
	Literal	1045	1045
	Literal	1057	1057
	Literal	1068	1068
DS$K_TYPE_DS_PROMPT	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_DS_START  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_ERRDEV  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_ERRHARD  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_ERROR_BODY  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
```

Symbol	Type	Defined	Referenced ...
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_ERROR_END	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_ERRPREP  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
```

Symbol	Type	Defined	Referenced ...
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_ERRSOFT	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
    Literal    1057
    Literal    1068
DS$K_TYPE_ERRSUP  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_ERRSYS  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
```

Symbol	Type	Defined	Referenced ...
	Literal		270
	Literal		284
	Literal		289
	Literal		294
	Literal		368
	Literal		375
	Literal		377
	Literal		381
	Literal		382
	Literal		398
	Literal		454
	Literal		628
	Literal		635
	Literal		645
	Literal		648
	Literal		650
	Literal		651
	Literal		666
	Literal		873
	Literal		880
	Literal		881
	Literal		882
	Literal		883
	Literal		884
	Literal		888
	Literal		910
	Literal		986
	Literal		1027
	Literal		1041
	Literal		1041
	Literal		1045
	Literal		1057
	Literal		1068
DS$K_TYPE_ERR_HALT	Literal		225
	Literal		227
	Literal		238
	Literal		243
	Literal		248
	Literal		254
	Literal		255
	Literal		260
	Literal		265
	Literal		270
	Literal		284
	Literal		289
	Literal		294
	Literal		368
	Literal		375
	Literal		377
	Literal		381
	Literal		382
	Literal		398
	Literal		454

Symbol	Type	Defined	Referenced ...
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_EXCEPTION	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	

Symbol	Type	Defined	Referenced ...
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_EXCEPTION_HEAD	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	

Symbol	Type	Defined	Referenced ...
DS$K_TYPE_FIRST_PASS	Literal	225	
	Literal		227
	Literal		238
	Literal		243
	Literal		248
	Literal		254
	Literal		255
	Literal		260
	Literal		265
	Literal		270
	Literal		284
	Literal		289
	Literal		294
	Literal		368
	Literal		375
	Literal		377
	Literal		381
	Literal		382
	Literal		398
	Literal		454
	Literal		628
	Literal		635
	Literal		645
	Literal		648
	Literal		650
	Literal		651
	Literal		666
	Literal		873
	Literal		880
	Literal		881
	Literal		882
	Literal		883
	Literal		884
	Literal		888
	Literal		910
	Literal		986
	Literal		1027
	Literal		1041
	Literal		1041
	Literal		1045
	Literal		1057
	Literal		1068
DS$K_TYPE_GENERAL	Literal	225	
	Literal		227
	Literal		238
	Literal		243
	Literal		248
	Literal		254
	Literal		255
	Literal		260
	Literal		265
	Literal		270
	Literal		284

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_GENERAL_ERROR  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
```

Symbol	Type	Defined	Referenced ...
	Literal 645		
	Literal 648		
	Literal 650		
	Literal 651		
	Literal 666		
	Literal 873		
	Literal 880		
	Literal 881		
	Literal 882		
	Literal 883		
	Literal 884		
	Literal 888		
	Literal 910		
	Literal 986		
	Literal 1027		
	Literal 1041		
	Literal 1041		
	Literal 1045		
	Literal 1057		
	Literal 1068		
DS$K_TYPE_NO_TESTS	Literal 225		
	Literal 227		
	Literal 238		
	Literal 243		
	Literal 248		
	Literal 254		
	Literal 255		
	Literal 260		
	Literal 265		
	Literal 270		
	Literal 284		
	Literal 289		
	Literal 294		
	Literal 368		
	Literal 375		
	Literal 377		
	Literal 381		
	Literal 382		
	Literal 398		
	Literal 454		
	Literal 628		
	Literal 635		
	Literal 645		
	Literal 648		
	Literal 650		
	Literal 651		
	Literal 666		
	Literal 873		
	Literal 880		
	Literal 881		
	Literal 882		
	Literal 883		
	Literal 884		

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_PARAM_ERROR  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_PROGRAM_END  Literal     225
    Literal     227
```

Symbol	Type	Defined	Referenced ...
	Literal		238
	Literal		243
	Literal		248
	Literal		254
	Literal		255
	Literal		260
	Literal		265
	Literal		270
	Literal		284
	Literal		289
	Literal		294
	Literal		368
	Literal		375
	Literal		377
	Literal		381
	Literal		382
	Literal		398
	Literal		454
	Literal		628
	Literal		635
	Literal		645
	Literal		648
	Literal		650
	Literal		651
	Literal		666
	Literal		873
	Literal		880
	Literal		881
	Literal		882
	Literal		883
	Literal		884
	Literal		888
	Literal		910
	Literal		986
	Literal		1027
	Literal		1041
	Literal		1041
	Literal		1045
	Literal		1057
	Literal		1068
DS$K_TYPE_PROGRAM_INFO	Literal		225
	Literal		227
	Literal		238
	Literal		243
	Literal		248
	Literal		254
	Literal		255
	Literal		260
	Literal		265
	Literal		270
	Literal		284
	Literal		289
	Literal		294

Symbol	Type	Defined	Referenced ...
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_PROGRAM_START	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_QIO_INVADP  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     550
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_QIO_NODRIVER  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_QIO_WRONGVER  Literal     225
    Literal     227
    Literal     238
    Literal     243
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_SCRIPT_ECHO  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
```

Symbol	Type	Defined	Referenced ...
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_SCRIPT_PNF	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_SCRIPT_PROMPT  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
```

Symbol	Type	Defined	Referenced ...
	Literal		1041
	Literal		1041
	Literal		1045
	Literal		1057
	Literal		1068
DS$K_TYPE_SCRIPT_SKIP	Literal		225
	Literal		227
	Literal		238
	Literal		243
	Literal		248
	Literal		254
	Literal		255
	Literal		260
	Literal		265
	Literal		270
	Literal		284
	Literal		289
	Literal		294
	Literal		368
	Literal		375
	Literal		377
	Literal		381
	Literal		382
	Literal		398
	Literal		454
	Literal		628
	Literal		635
	Literal		645
	Literal		648
	Literal		650
	Literal		651
	Literal		666
	Literal		873
	Literal		880
	Literal		881
	Literal		882
	Literal		883
	Literal		884
	Literal		888
	Literal		910
	Literal		986
	Literal		1027
	Literal		1041
	Literal		1041
	Literal		1045
	Literal		1057
	Literal		1068
DS$K_TYPE_SEQUENCE_ERROR	Literal		225
	Literal		227
	Literal		238
	Literal		243
	Literal		248
	Literal		254

Symbol	Type	Defined	Referenced ...
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_START_ERR	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- - --------------
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_START_LIST  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
```

Symbol	Type	Defined	Referenced ...
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	
	Literal	1045	
	Literal	1057	
	Literal	1068	
DS$K_TYPE_SUMMARY	Literal	225	
	Literal	227	
	Literal	238	
	Literal	243	
	Literal	248	
	Literal	254	
	Literal	255	
	Literal	260	
	Literal	265	
	Literal	270	
	Literal	284	
	Literal	289	
	Literal	294	
	Literal	368	
	Literal	375	
	Literal	377	
	Literal	381	
	Literal	382	
	Literal	398	
	Literal	454	
	Literal	628	
	Literal	635	
	Literal	645	
	Literal	648	
	Literal	650	
	Literal	651	
	Literal	666	
	Literal	873	
	Literal	880	
	Literal	881	
	Literal	882	
	Literal	883	
	Literal	884	
	Literal	888	
	Literal	910	
	Literal	986	
	Literal	1027	
	Literal	1041	
	Literal	1041	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ------- --------
    Literal    1045
    Literal    1057
    Literal    1068
DS$K_TYPE_USER_PROMPT  Literal     225
    Literal     227
    Literal     238
    Literal     243
    Literal     248
    Literal     254
    Literal     255
    Literal     260
    Literal     265
    Literal     270
    Literal     284
    Literal     289
    Literal     294
    Literal     368
    Literal     375
    Literal     377
    Literal     381
    Literal     382
    Literal     398
    Literal     454
    Literal     628
    Literal     635
    Literal     645
    Literal     648
    Literal     650
    Literal     651
    Literal     666
    Literal     873
    Literal     880
    Literal     881
    Literal     882
    Literal     883
    Literal     884
    Literal     888
    Literal     910
    Literal     986
    Literal    1027
    Literal    1041
    Literal    1041
    Literal    1045
    Literal    1057
    Literal    1068
DSA$AL_APTMAIL    Bind        155   155
DSA$GL_APTCOM     Bind        155
DSA$GL_DEVLEN     Bind        155
DSA$GL_DEVNAM     Bind        155
DSA$GL_ERRNO      Bind        155
DSA$GL_EVENT      Bind        155
DSA$GL_FLAGS      Bind        155   374     386     407     443     447     454     460     514     515     516
                  517    1006    1007    1010    1011    1049    1050
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DSA$GL_MSGPTR   Bind       155
DSA$GL_MSGTYP   Bind       155
DSA$GL_PASSES   Bind       155
DSA$GL_PASSNO   Bind       155
DSA$GL_SECTNO   Bind       155
DSA$GL_SID   Bind       155
DSA$GL_SUBTNO   Bind       155
DSA$GL_TESTNO   Bind       155
DSA$GL_UNITS   Bind       155
DSA$V_BINARY   Macro    Lib02   517
DSA$V_CRD_AUTOTEST_OFF  Macro    Lib02   386     407     460     514
DSA$V_CRD_MENUTEST_OFF  Macro    Lib02   374     443     515
DSA$V_CRD_MENUTEST_ON  Macro    Lib02   447     516
DSA$V_OPER   Macro    Lib02  1007    1011    1050
DSA$V_PROMPT   Macro    Lib02  1006    1010    1049
DSR$CHECK_AUTOTEST_OFF_SET BindRout   216   218c
DSR$CHECK_MENUTEST_OFF_SET BindRout   215   220c
ESSENTIAL_DEVICE  Register   700   708=    743.    746.
EXIT_ADDRESS   Local      485   487=    506ac
FALSE    Literal  Lib03   176     228     587     590     591     736     744     745     762    1004
            1048
FINISHED   Register   701   707=    719.
GENERIC_DEV_NAME  Local      800   804=    848a.
GET1ST_    Macro    Lib04   154     156     225     227     238     243     248     254     255     260
             265     270     284     289     294     368     375     377     381     382
             398     454     628     635     645     648     650     651     666     873
             880     881     882     883     884     888     910     986    1027    1041
            1045    1057    1068
GET2ND     Unbound          154     156     225     227     238     243     248     254     255     260
             265     270     284     289     294     368     375     377     381     382
             398     454     628     635     645     648     650     651     666     873
             880     881     882     883     884     888     910     986    1027    1041
            1045    1057    1068
HARDWARE_SUPPORT_VECTOR  Structur Lib03   323     798
HDWR_NAME_PTR   Register   595   786=    813=    817=    821=    825=    829=    833=    837=    841=    844=
             881.
HP$K_LENGTH   Literal  Lib02   600
HP$T_TYPE   Field    Lib02   786a
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal    154
HS$K_ACTION_ADVANCE_GENERAL_COM Literal    154
HS$K_ACTION_ADVANCE_STATE Literal    154
HS$K_ACTION_CHANGE_TABLE Literal    154
HS$K_ACTION_DIAGNOSTIC_ERROR Literal    154
HS$K_ACTION_DONE_GENERAL_COMS Literal    154
HS$K_ACTION_DO_NOTHING  Literal    154
HS$K_ACTION_DUMMY_3  Literal    154
HS$K_ACTION_DUMMY_4  Literal    154
HS$K_ACTION_DUMMY_5  Literal    154
HS$K_ACTION_DUMMY_6  Literal    154
HS$K_ACTION_INIT_HS  Literal    154
HS$K_ACTION_INTERNAL_ERROR Literal    154
HS$K_ACTION_LAST_ACTION  Literal    154
HS$K_ACTION_LOAD_ERROR  Literal    154
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_ACTION_LOAD_GENERAL_COM Literal     154
HS$K_ACTION_LOAD_LOAD_COM Literal     154
HS$K_ACTION_LOAD_SELECT_COM Literal     154
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     154
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     154
HS$K_ACTION_LOAD_START_COM Literal     154
HS$K_ACTION_PREPARATION_ERROR Literal     154
HS$K_ACTION_START_ERROR  Literal     154
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     154
HS$K_STATE_ADVANCE_GENERAL_COM Literal     154
HS$K_STATE_CHANGE_TABLE  Literal     154
HS$K_STATE_DIAGNOSTIC_ERROR Literal     154
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     154
HS$K_STATE_DONE_GENERAL_COMS Literal     154
HS$K_STATE_DUMMY_1  Literal     154
HS$K_STATE_DUMMY_2  Literal     154
HS$K_STATE_DUMMY_3  Literal     154
HS$K_STATE_DUMMY_4  Literal     154
HS$K_STATE_DUMMY_6  Literal     154
HS$K_STATE_INIT_HS  Literal     154
HS$K_STATE_INTERNAL_ERROR Literal     154
HS$K_STATE_LAST_STATE  Literal     154
HS$K_STATE_LOAD_ERROR  Literal     154
HS$K_STATE_LOAD_GENERAL_COM Literal     154
HS$K_STATE_LOAD_LOAD_COM Literal     154
HS$K_STATE_LOAD_SELECT_COM Literal     154
HS$K_STATE_LOAD_SET_EVENT_COM Literal     154
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     154
HS$K_STATE_LOAD_START_COM Literal     154
HS$K_STATE_PREPARATION_ERROR Literal     154
HS$K_STATE_START_ERROR  Literal     154
I     Local       638    643.    663.    760.    867.    886.    907.
J     Local       806    809.
JSB_BEGIN   Linkage  Lib03   523
JSB_NONE    Linkage  Lib01   215     216     506
LOCAL_OPERINPUT_BUFFER  Local       952  1031a   1038.
MEN$GT_INVOPERINPUT  External Lib03  1041a
MEN$GT_MENFUNCSUP_FILE  External Lib03   238a    243a
MEN$GT_OPERATOR_ABORT  External Lib03   284a
MEN$GT_OPERATOR_STOP  External Lib03   270a
MEN$GT_PRESS_RETURN_TM  External Lib03  1019a
MEN$GT_SUPPORT_FILE_LOAD_ERR External Lib03   243a
MEN$GT_SUPPORT_FILE_NOT_FOUND External Lib03   238a
MEN$GT_TM_MSG   External Lib03  1021a
MEN$K_HS_STATE_TABLE  Literal     154
MEN$K_MM_STATE_TABLE  Literal     154
MEN$K_OPERINPUT_BUFFER_SIZE Literal  Lib03   952
MEN$K_SUPBLK_SIZE  Literal  Lib03   599
MEN$K_TQ_STATE_TABLE  Literal     154
MEN$MENU_CLEANUP  ExtRout  Lib03   445c    449c
MENU$K_EXIT_AUTOSIZER_ERROR Literal     154   372
MENU$K_EXIT_INTERNAL_ERROR Literal     154   251
MENU$K_EXIT_LAST_REASON  Literal     154
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------------- ----- -------- ---------------
MENU$K_EXIT_OPERATOR_ABORT Literal     154   273
MENU$K_EXIT_OPERATOR_STOP Literal     154   268
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     154   241
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     154   236
MM$K_ACTION_ADVANCE_STATE Literal     154
MM$K_ACTION_AUTOSIZER_ERROR Literal     154
MM$K_ACTION_BUILD_DDBS  Literal     154
MM$K_ACTION_BUILD_HSTS  Literal     154
MM$K_ACTION_CHANGE_TABLE Literal     154
MM$K_ACTION_DONE_INITS  Literal     154
MM$K_ACTION_DO_NOTHING  Literal     154
MM$K_ACTION_DUMMY_3  Literal     154
MM$K_ACTION_DUMMY_4  Literal     154
MM$K_ACTION_DUMMY_5  Literal     154
MM$K_ACTION_DUMMY_6  Literal     154
MM$K_ACTION_DUMMY_7  Literal     154
MM$K_ACTION_DUMMY_8  Literal     154
MM$K_ACTION_INTERNAL_ERROR Literal     154
MM$K_ACTION_LAST_ACTION  Literal     154
MM$K_ACTION_LOAD_AUTOSIZER Literal     154
MM$K_ACTION_MENU_PROMPT  Literal     154
MM$K_ACTION_PRINT_MENU  Literal     154
MM$K_ACTION_PRINT_MENU_HEADING Literal     154
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     154
MM$K_ACTION_START_AUTOSIZER Literal     154
MM$K_STATE_AUTOSIZER_ERROR Literal     154
MM$K_STATE_AUTOSIZER_RUNNING Literal     154
MM$K_STATE_BUILD_DDBS  Literal     154
MM$K_STATE_BUILD_HSTS  Literal     154
MM$K_STATE_CHANGE_TABLE  Literal     154
MM$K_STATE_DONE_INITS  Literal     154
MM$K_STATE_DUMMY_1  Literal     154
MM$K_STATE_DUMMY_2  Literal     154
MM$K_STATE_DUMMY_3  Literal     154
MM$K_STATE_DUMMY_5  Literal     154
MM$K_STATE_DUMMY_7  Literal     154
MM$K_STATE_DUMMY_8  Literal     154
MM$K_STATE_INTERNAL_ERROR Literal     154
MM$K_STATE_LAST_STATE  Literal     154
MM$K_STATE_LOAD_AUTOSIZER Literal     154
MM$K_STATE_MENU_PROMPT  Literal     154
MM$K_STATE_PRINT_MENU  Literal     154
MM$K_STATE_PRINT_MENU_HEADING Literal     154
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     154
MM$K_STATE_START  Literal     154
MM$K_STATE_START_AUTOSIZER Literal     154
MODNAM    Macro    Lib01   172
NEXT_PTR   Register   410   432=    437.
NUL2ND_    Macro    Lib04   154     156     225     227     238     243     248     254     255     260
              265     270     284     289     294     368     375     377     381     382
              398     454     628     635     645     648     650     651     666     873
              880     881     882     883     884     888     910     986    1027    1041
             1045    1057    1068
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
NUMB_BAD_DEVICES  Local      304   314=    329=    329.    335.
OFF    Literal  Lib03   514     516
ON    Literal  Lib03   515     517    1010    1011
OPER    Register   947  1007=   1050.
PREP_ERROR   Register   698   710=    721.    726.
PRINT_EXIT_TO   Routine    212   405c
PROMPT    Register   946  1006=   1049.
PROMPT_MSG   Local      950  1015=   1017=   1019=   1021=   1031.
PTABLE_PTR   Register   600   775=    783.    786aa   793.
SCREEN$K_CLEAR_SCREEN  Literal    156  1055
SCREEN$K_COUNT_LINE  Literal    156  1061
SCREEN$K_PRESS_RETURN  Literal    156   996    1015
SCREEN$K_PRESS_RETURN_MM Literal    156   467    996    1017
SCREEN$K_PRESS_RETURN_TM Literal    156   997   1019
SCREEN$K_TM_MSG   Literal    156   997    1021
SCREEN_COUNTER   Register   597   657=    667=    870.    874=    877=    877.
SCREEN_OPERATION  RoutForm   918   994.   1013.
SIZE    Local      424   427=    434.
SOME_FAIL   Local      587   587=    645.    754=    754.
SOME_INC_PREP   Local      590   590=    650.    756=    756.
SOME_OTHER   Local      591   591=    651.    758=    758.    905.
STATUS    Local     1031  1031=   1031.
STOP_REASON   RoutForm   184   234.    387.    388.    389.    390.    391.    401.    402.    403.    461.
             462.    463.    479.    480.    481.
SUPBLK_PTR   Register   599
TEMP_DEV_NAME   Local      801   812=    816=    820=    824=    828=    832=    836=    840=    849a.
TERMCHAR$B_TYPE   Field    Lib03   619.    621.    622.    972.
TQ$K_ACTION_ADVANCE_STATE Literal    154
TQ$K_ACTION_CHANGE_TABLE Literal    154
TQ$K_ACTION_DONE_TEST_QUEUE Literal    154
TQ$K_ACTION_DO_NOTHING  Literal    154
TQ$K_ACTION_DUMMY_3  Literal    154
TQ$K_ACTION_DUMMY_4  Literal    154
TQ$K_ACTION_DUMMY_5  Literal    154
TQ$K_ACTION_GET_DDB  Literal    154
TQ$K_ACTION_INIT_TQ  Literal    154
TQ$K_ACTION_INTERNAL_ERROR Literal    154
TQ$K_ACTION_LAST_ACTION  Literal    154
TQ$K_STATE_CHANGE_TABLE  Literal    154
TQ$K_STATE_DONE_TEST_QUEUE Literal    154
TQ$K_STATE_DUMMY_1  Literal    154
TQ$K_STATE_DUMMY_2  Literal    154
TQ$K_STATE_DUMMY_3  Literal    154
TQ$K_STATE_DUMMY_4  Literal    154
TQ$K_STATE_DUMMY_5  Literal    154
TQ$K_STATE_GET_DDB  Literal    154
TQ$K_STATE_INIT_TQ  Literal    154
TQ$K_STATE_INTERNAL_ERROR Literal    154
TQ$K_STATE_LAST_STATE  Literal    154
TRUE    Literal  Lib03   518    1009
TT$_VT100   Literal  Lib04   619     975
TT$_VT200_SERIES  Literal  Lib04   622     977
TT$_VT52   Literal  Lib04   621     974
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
TYPE_CLEAR_SCREEN  Local       949   974=    975=    977=   1045.   1057.
T_PROMPT_DEFAULT  Bind        956  1031a   1039.
VALID_STATUS   Local       951  1004=   1038=   1043.
VECTOR    Local       501   503.



  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDSTOP.B32;1
    2  Module CRDSTOP
  128  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
  131  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
  134  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
  137  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
 1074  Eludom CRDSTOP


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     a Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;                --- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671         2          0        46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923        28          3        94          00:00.2
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486       105         21        62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          6           0      1000          00:04.2
```

```
;     COMMAND QUALIFIERS

; BLISS CRDSTOP.B32/LIST=CRDSTOP.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDSTOP.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  2066 code + 309 data bytes
; Run Time:    01:44.7
; Elapsed Time:    03:55.7
; Lines/CPU Min:     615
; Lexemes/CPU-Min: 73406
; Memory Used: 542 pages
; Compilation Complete
```

```
;    0001  0 %TITLE '*** CRDTEXT Module'
;    0002  0 MODULE CRDTEXT (    ! Contains all of the CRD output ASCIC text
;    0003  0    IDENT = '11'
;    0004  0    ) =
;    0005  0 !++
;    0006  0 ! COPYRIGHT (c) 1980, 1981, 1985
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the CRD-Loadable image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! This module contains all of the messages (ASCIC text) that get printed
;    0032  0 !       by CRD-AutoTest.
;    0033  0 !
;    0034  0 ! Author:
;    0035  0 !
;    0036  0 ! Jack Stansbury
;    0037  0 !
;    0038  0 ! Creation Date:
;    0039  0 !
;    0040  0 ! May 7, 1982
;    0041  0 !
;    0042  0 ! Version:
;    0043  0 !
;    0044  0 ! 6.9
;    0045  0 !
;    0046  0 ! Modified By:
;    0047  0 !
;    0048  0 ! 01A Jack Stansbury, Version 1.0, July 9, 1982
;    0049  0 !  Added the CRD$GT_Prep_Error_Spaces_FAO message string. This can be taken out
;    0050  0 !  when the DS's FAO routine is fixed.
;    0051  0 ! 01B  John Ciukaj Jan. 1983
;    0052  0 !  - Enhancements for Release 13 CRD Menu version 1.2.
;    0053  0 !
;    0054  0 ! 02A Jack Stansbury, Version 1.1, July 25, 1982
;    0055  0 !  Added the CRD$GT_Menu and CRD$GT_Auto strings so that the outputting messages can
;    0056  0 !  be for the appropriate test name.
;    0057  0 ! 02B Scott Cohen June 24, 1983 Version 1.2
;    0058  0 !  For soft console support
;    0059  0 !  - Added MEN$GT_End_of_List,
;    0060  0 !  - MEN$GT_TM_Pause,
;    0061  0 !
;    0062  0 ! 03A Jack Stansbury, Version 1.1, September 12, 1982
;    0063  0 !  Added the CRD$GT_Exit To_CLI message string.
;    0064  0 ! 03B     Ron Parker June 1, 1984 Version 1.5
;    0065  0 !  - Bumped version number from 1.4 to 1.5
;    0066  0 !
;    0067  0 ! 04A Jack Stansbury, Version 1.1, September 15, 1982
;    0068  0 !  Added the CRD$GT_DS_Command_Out string for outputting the DS command when
;    0069  0 !  CRD sticks one in the buffer. This happens when the DSA$V_CRD_Trace bit is set.
;    0070  0 ! 04B Ron Parker Octorber 15, 1984
;    0071  0    - Added device prep message #7
;    0072  0 .
;    0073  0 . 05A Jack Stansbury, Version 1.1, September 24, 1982
;    0074  0 !  Added the CRD$GT_Error_Complete message, which should never get printed!
;    0075  0 ! 05B Ron Parker October 26,1984
;    0076  0 !  - Added VMS error message
;    0077  0 !
;    0078  0 . 06A John Ciukaj Version 1.2 8-Apr 1983
```

```
;    0079  0 !  - Added  CRD$GT_Inv_Base_System
;    0080  0 !    CRD$GT_CRD_Fatal_Error
;    0081  0 !  - Replaced the 'no errors' text in CRD$GT_CRD_Incomplete
;    0082  0 !  - Deleted trailimg '***' in CRD$GT_Prep_Error_Spaces
;    0083  0 !
;    0084  0 ! 07A Scott Cohen Version 1.2 9-Jun-1983
;    0085  0 !  For soft console support
;    0086  0 !  - Added CRD$GT_Press_Return,
;    0087  0 !  -  CRD$GT_Clear_Screen,
;    0088  0 !  -  all strings output by AutoTest Summary
;    0089  0 !
;    0090  0 ! 08      Ron Parker  31-OCT-1984
;    0091  0 !  - Combined both ASCIC files into one.
;    0092  0 !
;    0093  0 ! 09 Ron Parker 27-FEB-1985
;    0094  0 !  - Changed reference to 11/730,725 to VAX-11 CRD
;    0095  0 !
;    0096  0 ! 10 Amy Iorio 08-JULY-1985
;    0097  0 !  - Changed wording of TESTPREP statements.
;    0098  0 !
;    0099  0 ! 11 Amy Iorio 15-JULY-1985
;    0100  0 !  - Deleted crd$gt_clear_screen(_vt52) in order to
;    0101  0 !  put their equivalent values directly in CRDSTOP.
;    0102  0 !
;    0103  0 ! 12 Amy Iorio 04-SEPT-1985
;    0104  0 !  - Bumped version number to 2.
;    0105  0 !
```

```
;    0106  0 %SBTTL 'Libraries'
;    0107  1 BEGIN      ! Begin the CRDTEXT module
;    0108  1
;    0109  1 LIBRARY
;    0110  1  '$DS';     ! Use DS.L32 first
;    0111  1
;    0112  1 LIBRARY
;    0113  1  '$Diag';   ! Use Diag.L32 second
;    0114  1
;    0115  1 LIBRARY
;    0116  1  '$CRD';    ! Use CRD.L32 third
;    0117  1
;    0118  1 LIBRARY
;    0119  1  'Sys$Library:Lib';  ! Use Lib as last resort
;    0120  1
;    0121  1
;    0122  1
;    0123  1
;    0124  1 %SBTTL 'Included Macros and Literals'
;    0125  1
;    0126  1  $CRD_Literals;   ! Define the literals for CRd
;    0127  1  $MEN_Literals;   ! Define the literals for MEn
;    0128  1
;    0129  1
;    0130  1
;    0131  1
;    0132  1 %SBTTL 'Psect Definitions'
;    0133  1
;    0134  1 PSECT
;    0135  1      PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0136  1
;    0137  1 PSECT
;    0138  1      CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0139  1
;    0140  1 PSECT
;    0141  1      OWN     - Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0142  1
;    0143  1 PSECT
;    0144  1      GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING MODE (LONG_RELATIVE));
```

```
;    0145  1 %SBTTL 'Module-Global Static Storage'
;    0146  1
;    0147  1  ModNam ('CRDTEXT');  ! Module name for ErrSup macro
;    0148  1
;    0149  1
;    0150  1
;    0151  1
;    0152  1 %SBTTL 'Global Own Storage'
;    0153  1
;    0154  1 GLOBAL
;    0155  1  CRD$GA_CRD_Message_Buffer : VECTOR [CRD$K_CRD_Message_Buffer_Len, BYTE];
;    0156  1
```

```
;    0157  1 %SBTTL 'Global Bindings'
;    0158  1
;    0159  1 GLOBAL BIND
;    0160  1  !+
;    0161  1  ! These are auxilliary strings.
;    0162  1  !-
;    0163  1
;    0164  1  CRD$GT_Space_And_Space  = $ASCIC (' AND '),
;    0165  1  CRD$GT_New_Line   = $ASCIC ('!/'),
;    0166  1  CRD$GT_Tab   = $ASCIC ('!_'),     ! [02]
;    0167  1  CRD$GT_Slash   = $ASCIC ('/'),     ! [02]
;    0168  1  CRD$GT_Unknown   = $ASCIC ('????'),     ! [02]
;    0169  1  CRD$GT_2_Tabs   = $ASCIC ('!_!_'),     ! [02]
;    0170  1 ![11] CRD$GT_Clear_Screen  = $ASCIC ('[H[J'),   ! [02]
;    0171  1 ![11] CRD$GT_Clear_Screen_VT52 = $ASCIC ('HJ'),   ! [02]
;    0172  1
;    0173  1
;    0174  1  !+          ! [02]
;    0175  1  ! These strings define the names of the two types of CRD.   ! [02]
;    0176  1  !-          ! [02]
;    0177  1
;    0178  1  CRD$GT_Menu   = $ASCIC ('MENU'),     ! [02]
;    0179  1  CRD$GT_Auto   = $ASCIC ('AUTO'),     ! [02]
;    0180  1
;    0181  1  !+
;    0182  1  ! These strings are always output before a diagnostic is started.
;    0183  1  !-
;    0184  1
;    0185  1  CRD$GT_Testing_Device      = $ASCIC ('!/!18AC'),
;    0186  1  CRD$GT_Testing_Started      = $ASCIC (' TESTING STARTED - RUN TIME ='),
;    0187  1  CRD$GT_RunTime       = $ASCIC (' !4AC MINUTES'),
;    0188  1  CRD$GT_Fail       = $ASCIC ('   *FAILED*!/'),
;    0189  1  CRD$GT_Pass       = $ASCIC ('    PASSED !/'),
;    0190  1
;    0191  1  !+
;    0192  1  ! These lines are output before the last two lines are output. They summarize what has happened
;    0193  1  ! and what should be done by the user.
;    0194  1  !-
;    0195  1
;    0196  1  CRD$GT_CRD_No_Errors     = $ASCIC ('!/!/!AC TEST COMPLETE - NO ERRORS'),    ![02]
;    0197  1  CRD$GT_CRD_Errors_Complete = $ASCIC ('!/!/!AC TEST COMPLETE - ERRORS'),    ![05]
;    0198  1  CRD$GT_CRD_Bad_Device     = $ASCIC ('!/!/!AC TEST INCOMPLETE - !AC BAD - CALL FIELD SERVICE'), ![02]
;    0199  1  CRD$GT_CRD_Incomplete  =
; P  0200  1   $ASCIC ('!/!/!AC TEST INCOMPLETE - ',
;    0201  1     'IF ASSISTANCE NEEDED, CALL FIELD SERVICE'),          ![06]
; P  0202  1  CRD$GT_CRD_Dev_UnTested  = $ASCIC ('!/!/!AC TEST INCOMPLETE',      ![02]
;    0203  1          '!/CHECK DEVICE PREPARATION - IF OK, THEN CALL FIELD SERVICE'),
;    0204  1  CRD$GT_CRD_EVSBA_Error  = $ASCIC ('!/!/!AC TEST ABORTED - CALL FIELD SERVICE'),   ![02]
;    0205  1  CRD$GT_CRD_EVSBB_Error  = $ASCIC ('!/!/!AC TEST ABORTED - CALL FIELD SERVICE'),   ![02]
;    0206  1  CRD$GT_CRD_Fatal_Error  = $ASCIC ('!/!/!AC TEST ABORTED - CALL FIELD SERVICE'),   ![06]
;    0207  1  CRD$GT_Internal_Error_Abort = $ASCIC ('!/!/!AC TEST ABORTED - CALL FIELD SERVICE'),   ![02]
;    0208  1  CRD$GT_Control_C_Abort      = $ASCIC ('!/!/!AC TEST ABORTED - CONTROL-C TYPED AT TERMINAL'), ![02]
;    0209  1
;    0210  1  !+
;    0211  1  ! These two messages are always output when CRD-AutoTest exits.
```

```
;    0212  1  !-
;    0213  1
;    0214  1  CRD$GT_CRD_End_Message   = $ASCIC ('!/**** END OF VAX-11 CRD !AC TEST ****'),   ![02]
;    0215  1  CRD$GT_Exit_To_Console      = $ASCIC ('!/!/RETURNING TO VAX-11 CRD CONSOLE, WAIT FOR  >>> PROMPT!/'),
;    0216  1  CRD$GT_Exit_To_CLI  = $ASCIC ('!/!/RETURNING TO VAX DIAGNOSTIC SUPERVISOR COMMAND MODE!/'), ![03]
;    0217  1
;    0218  1  !+
;    0219  1  ! Soft console support messages - the operator must press return to continue
;    0220  1  !-
;    0221  1
;    0222  1  CRD$GT_Press_Return  = $ASCIC ('Press RETURN for more...'),
;    0223  1  CRD$GT_Press_Return_MM  = $ASCIC ('Press RETURN for MAIN MENU...'),
```

```
;    0224  1
;    0225  1  !+
;    0226  1  ! AUTOSIZER error messages
;    0227  1  !-
;    0228  1  CRD$GT_Inv_Base_System  = $ASCIC ('!/NO SUPPORTED DISK CONTROLLER FOUND BY CRD MACRO TEST'), ![06]
; P  0229  1  CRD$GT_AutoSizer_Error  = $ASCIC ('!/UNABLE TO DETERMINE DEVICES ON THE SYSTEM ',
;    0230  1          '- NO FURTHER TESTING POSSIBLE!/'),
;    0231  1  !+
;    0232  1  ! These messages are output after the  TESTING STARTED  message has been printed (instead of
;    0233  1  ! the 'PASSED  or  FAILED  string). They indicate something is wrong with the current diagnostic.
;    0234  1  !-
;    0235  1
;    0236  1  CRD$GT_Dev_Unavailable  = $ASCIC ('!/DEVICE NOT AVAILABLE!/'),
;    0237  1  CRD$GT_Dev_Unavailable_2 = $ASCIC ('DEVICE NOT AVAILABLE!/'),
;    0238  1  CRD$GT_Cannot_Load_Diag  = $ASCIC ('!/DEVICE DIAGNOSTIC NOT FOUND!/'),
;    0239  1  CRD$GT_Start_Error  = $ASCIC ('!/CANNOT CONTINUE TESTING THIS DEVICE!/'),
;    0240  1 ! CRD$GT_Prep_Error_Spaces = $ASCIC ('!/** !#* !AC!#*'),        ![06]
;    0241  1  CRD$GT_Prep_Error_Spaces = $ASCIC ('!/*** !AC'),        ![07]
;    0242  1
;    0243  1  CRD$GT_Internal_Error  = $ASCIC ('!/!/THIS VERSION OF CRD !AC TEST IS CORRUPTED!/'),  ![02]
;    0244  1
;    0245  1  !+
;    0246  1  ! This string is used to output a DS command that was put into the CLI command buffer.   ![03]
;    0247  1  ! This is primarily used for tracing CRD.        ![03]
;    0248  1  !-
;    0249  1
;    0250  1  CRD$GT_DS_Command_Out  = $ASCIC ('!/*** DS Command: !60<!AD!>!/'),    ![03]
;    0251  1
;    0252  1  !+
;    0253  1  ! These strings are output by the $CRD_Message macro.
;    0254  1  !-
;    0255  1
;    0256  1  CRD$GT_Same_Line_Message = $ASCIC (''),
;    0257  1  CRD$GT_New_Line_Message  = $ASCIC ('!/'),
;    0258  1  CRD$GT_Star_Line_Prefix_Message = $ASCIC ('!/** '),
;    0259  1  CRD$GT_Star_Line_Suffix_Message = $ASCIC (' **'),
;    0260  1
;    0261  1  !+
;    0262  1  ! These strings are output by the Auto Test Summary.
;    0263  1  !-
;    0264  1  CRD$GT_Summary_Title =
;    0265  1    $ASCIC ('!/!/-- SUMMARY --!/'),
;    0266  1  CRD$GT_All_Failures   = $ASCIC ('!/    Failed:  (Hardware name/Generic name)!/'),
;    0267  1  CRD$GT_All_Passes   = $ASCIC ('!/    Passed:  (Hardware name/Generic name)!/'),
;    0268  1  CRD$GT_All_Inc_Hdwr_Prep  =
;    0269  1    $ASCIC ('!/    Incorrect Hardware Preparation:  (Hardware name/Generic name)!/'),
;    0270  1  CRD$GT_All_Others   = $ASCIC ('!/    Others:  (Diagnostic message)!/'),
;    0271  1  CRD$GT_Init_All_Passes = $ASCIC ('!_CPU!_!_!_MEMORY!_!_');
;    0272  1
```

```
;    0273  1 %TITLE '*** CRD MENU - Message Module'
;    0274  1 %SBTTL 'Global Bindings'
;    0275  1
;    0276  1 GLOBAL BIND
;    0277  1
;    0278  1 !+
;    0279  1 ! CRD MENU TEST Title with Version number
;    0280  1 !-
;    0281  1
;    0282  1 MEN$GT_MenuTest_Begin =
; P  0283  1  $ASCIC ('!/!/!19* CUSTOMER RUNNABLE DIAGNOSTIC PACKAGE',
; P  0284  1    '!/!/!30* Version   V020',
; P  0285  1    '!/!/!/!4** BEGIN VAX-11 CRD MENU TEST !4**',
; P  0286  1      '!/!/Type the CTRL key and the C key (together) at!/',
; P  0287  1    'any time to interrupt Menu Test processing',
; P  0288  1      '!/!/!/'
;    0289  1     ),
;    0290  1
;    0291  1 MEN$GT_AutoSizer_Begin =
;    0292  1  $ASCIC ('VAX-HARDWARE IDENTIFICATION - RUN TIME = 00:30 MINUTES!/!/Please wait . . .!/!/'),
;    0293  1
;    0294  1 MEN$GT_AutoSizer_End =
;    0295  1  $ASCIC ('VAX-11 CRD HARDWARE IDENTIFICATION COMPLETE!/'),
```

```
;    0296  1
;    0297  1 !+
;    0298  1 ! Functional Test Main Menu text
;    0299  1 !-
;    0300  1
;    0301  1 MEN$GT_FTMTitle =
;    0302  1  $ASCIC ('!/!/!27*-!/MAIN MENU - Functional Test!/!27*-'),
;    0303  1
;    0304  1 MEN$GT_FTMenu_Selection =
;    0305  1  $ASCIC ('!/    !AD = !AC'),
;    0306  1
;    0307  1 MEN$GT_FTMSel_Exit =
;    0308  1  $ASCIC ('Exit Menu Test'),
;    0309  1
;    0310  1 MEN$GT_FTMSel_Prep =
;    0311  1  $ASCIC ('Print preparation required for test of supported hardware'),
;    0312  1
;    0313  1 MEN$GT_FTMSel_Test =
;    0314  1  $ASCIC ('Select and Test hardware from TEST MENU'),
;    0315  1
;    0316  1 MEN$GT_FTMSEL_System_Hdwr =
;    0317  1  $ASCIC ('Print list of identified system hardware and support status'),
;    0318  1
;    0319  1 MEN$GT_FTMSEL_FncTst_All =
;    0320  1  $ASCIC ('TEST ALL Supported Hardware'),
;    0321  1
;    0322  1
;    0323  1 MEN$GT_FTM_Choice =
;    0324  1  $ASCIC ('!/!/    Type one of the above (for example, A), and press RETURN / /'),
;    0325  1
;    0326  1 MEN$GT_FTMPrompt =
;    0327  1  $ASCIC ('    Enter MAIN MENU choice: '),
```

```
;    0328  1
;    0329  1 !+
;    0330  1 ! TEST MENU Text
;    0331  1 !-
;    0332  1
;    0333  1 MEN$GT_TMTitle =
;    0334  1  $ASCIC ('!/!/!9*-!/TEST MENU!/!9*-'),
;    0335  1
;    0336  1 MEN$GT_TMenu_Selection =
;    0337  1  $ASCIC ('    !3<!AD!UL!> = !9AC !9AC !6AC'),
;    0338  1
;    0339  1 MEN$GT_TMenu_TstAll =
;    0340  1  $ASCIC ('    !3<!AD!UL!> = TEST ALL of the above supported hardware'),
;    0341  1
;    0342  1 MEN$GT_TM_Choice =
;    0343  1  $ASCIC ('!/!/    Type one of the above (for example, A1), and press RETURN to start testing!/!/'),
;    0344  1
;    0345  1 MEN$GT_TMPrompt =
;    0346  1    $ASCIC ('    Enter TEST MENU choice: '),
;    0347  1
;    0348  1 MEN$GT_TM_Msg =
;    0349  1    $ASCIC ('*** Press RETURN for remainder of TEST MENU choices ***'),
;    0350  1
;    0351  1 MEN$GT_Press_Return_TM =
;    0352  1    $ASCIC ('Press RETURN for TEST MENU...'),
```

```
;    0353  1 !+
;    0354  1 ! Control C Menu Text
;    0355  1 !-
;    0356  1
;    0357  1 MEN$GT_CntlC_Menu_Title =
;    0358  1  $ASCIC ('!/    !14*-!/    CONTROL-C MENU!/    !14*-!/'),
;    0359  1
;    0360  1 MEN$GT_CntlC_Menu_Selection =
;    0361  1  $ASCIC ('!/        !AD = !AC'),
;    0362  1
;    0363  1 MEN$GT_CntlC_Sel_Exit = MEN$GT_FTMSel_Exit,
;    0364  1
;    0365  1 MEN$GT_CntlC_Sel_Continue =
;    0366  1  $ASCIC ('Resume process interrupted by the CONTROL-C'),
;    0367  1
;    0368  1 MEN$GT_CntlC_Sel_FTMenu =
;    0369  1  $ASCIC ('Abort the current process, and return to the MAIN MENU'),
;    0370  1
;    0371  1 MEN$GT_CntlCM_Choice =
;    0372  1  $ASCIC ('!/!/        Type one of the above (for example, A) and press RETURN!/!/'),
;    0373  1
;    0374  1 MEN$GT_CntlC_Menu_Prompt =
;    0375  1  $ASCIC ('        Enter CONTROL-C MENU choice: '),
;    0376  1
;    0377  1 MEN$GT_CntlC_Continuing =
;    0378  1  $ASCIC ('!/Resuming the interrupted process!/'),
```

```
;    0379  1
;    0380  1 !+
;    0381  1 ! Test Class Names
;    0382  1 !-
;    0383  1
;    0384  1 MEN$GT_TstCla_CPU =
; L  0385  1    UPLIT BYTE (%ASCIC %STRING
;    0386  1   (%EXACTSTRING (MEN$K_TstTxt_TstCla_Sz,%C' ','CPU')),
;    0387  1
;    0388  1 MEN$GT_TstCla_Memory =
; L  0389  1    UPLIT BYTE (%ASCIC %STRING
;    0390  1   (%EXACTSTRING (MEN$K_TstTxt_TstCla_Sz,%C' ','MEMORY')),
;    0391  1
;    0392  1 MEN$GT_TstCla_Disk =
; L  0393  1  UPLIT BYTE (%ASCIC %STRING
;    0394  1   (%EXACTSTRING (MEN$K_TstTxt_TstCla_Sz,%C' ','DISK')),
;    0395  1
;    0396  1 MEN$GT_TstCla_Tape =
; L  0397  1  UPLIT BYTE (%ASCIC %STRING
;    0398  1   (%EXACTSTRING (MEN$K_TstTxt_TstCla_Sz,%C' ','TAPE')),
;    0399  1
;    0400  1 MEN$GT_TstCla_COMM =
; L  0401  1  UPLIT BYTE (%ASCIC %STRING
;    0402  1   (%EXACTSTRING (MEN$K_TstTxt_TstCla_Sz,%C' ','COMM.')),
;    0403  1
;    0404  1 MEN$GT_TstCla_REAL =
; L  0405  1  UPLIT BYTE (%ASCIC %STRING
;    0406  1   (%EXACTSTRING (MEN$K_TstTxt_TstCla_Sz,%C' ','REALTIME')),
;    0407  1
;    0408  1 MEN$GT_TstCla_TERM =
; L  0409  1  UPLIT BYTE (%ASCIC %STRING
;    0410  1   (%EXACTSTRING (MEN$K_TstTxt_TstCla_Sz,%C' ','TERMINAL')),
;    0411  1
;    0412  1 MEN$GT_TstCla_PRINTER =
; L  0413  1  UPLIT BYTE (%ASCIC %STRING
;    0414  1   (%EXACTSTRING (MEN$K_TstTxt TstCla_Sz,%C' ','PRINTER')),
;    0415  1
;    0416  1 MEN$GT_TstCla_CARD =
; L  0417  1  UPLIT BYTE (%ASCIC %STRING
;    0418  1   (%EXACTSTRING (MEN$K_TstTxt TstCla_Sz,%C' ','CARD')),
;    0419  1
;    0420  1 MEN$GT_TstCla_IOADAP =
; L  0421  1  UPLIT BYTE (%ASCIC %STRING
;    0422  1   (%EXACTSTRING (MEN$K_TstTxt_TstCla_Sz,%C' ','ADAPTOR')),
;    0423  1
;    0424  1 MEN$GT_TstCla_NullName =
; L  0425  1  UPLIT BYTE (%ASCIC %STRING
;    0426  1   (%EXACTSTRING (MEN$K_TstTxt TstCla Sz,%C' ',' ')),
;    0427  1
;    0428  1 MEN$GT_TstCla_AI  - MEN$GT TstCla NullName,
```

```
;   0429  1
;   0430  1 !+
;   0431  1 ! Print system hardware text
;   0432  1 !-
;   0433  1
;   0434  1 MEN$GT_SysHdwr_Title =
; P 0435  1  $ASCIC ('!/!8* !9<Hardware!> !8<Generic!> !9<Test Menu!>',
; P 0436  1      '!/!8* !9<Name!> !8<Name!> !9<Support!>',
; P 0437  1      '!/!8* !9*- !8*- !9*-./'
;   0438  1     ),
;   0439  1 MEN$GT_System_Hdwr_Text =
;   0440  1  $ASCIC ('!8* !9AC !8AC !AC'),
;   0441  1
;   0442  1 MEN$GT_Supported =
;   0443  1  $ASCIC ('SUPPORTED'),
;   0444  1
;   0445  1 MEN$GT_NoSupport =
;   0446  1  $ASCIC ('No Support'),
;   0447  1
;   0448  1 MEN$GT_End_of_List =
;   0449  1  $ASCIC ('!/!_!_[End of list]'),
```

```
;    0450  1 %SBTTL ' Info messages'
;    0451  1
;    0452  1 !+
;    0453  1 ! Functional Testing Title
;    0454  1 !-
;    0455  1
;    0456  1 MEN$GT_CtrlC_FncTst =
; P  0457  1  $ASCIC ('!/!/!4* Type the CTRL key and the C key (together)',
; P  0458  1    '!/!4* at any time to interrupt Functional Testing'
;    0459  1    ),
;    0460  1
;    0461  1
;    0462  1 MEN$GT_FuncTest_Begin =
;    0463  1  $ASCIC ('!/!/!/!4* !10** BEGIN FUNCTIONAL TESTING !10**'),
;    0464  1
;    0465  1 MEN$GT_Runtime_Total =
;    0466  1  $ASCIC ('!/!/!15* RUN TIME = !2ZL:!2ZL MINUTES!/!/'),
;    0467  1
;    0468  1 MEN$GT_FncTst_Cmpl_NoErr =
;    0469  1  $ASCIC ('!/FUNCTIONAL TESTING COMPLETE - NO ERRORS'),
;    0470  1
;    0471  1 MEN$GT_FncTst_InCmpl_Err =
;    0472  1  $ASCIC ('!/** FUNCTIONAL TESTING INCOMPLETE - ERRORS'),
;    0473  1
;    0474  1 MEN$GT_FncTst_Cmpl_Err =
;    0475  1  $ASCIC ('!/** FUNCTIONAL TESTING COMPLETE - ERRORS'),
;    0476  1
;    0477  1 MEN$GT_FncTst_InCmpl_NoErr =
;    0478  1  $ASCIC ('!/** FUNCTIONAL TESTING INCOMPLETE'),
;    0479  1
;    0480  1 MEN$GT_Assistance_Msg =
;    0481  1  $ASCIC ('!/** IF ASSISTANCE IS NEEDED - CALL FIELD SERVICE'),
;    0482  1
;    0483  1 MEN$GT_MenuTest_Aborted =
;    0484  1  $ASCIC ('!/!/VAX-11 CRD MENU TEST ABORTED'),
;    0485  1
;    0486  1 MEN$GT_Operator_Abort =
;    0487  1  $ASCIC ('!/!/VAX-11 CRD MENU TEST ABORTED BY OPERATOR'),
;    0488  1
;    0489  1 MEN$GT_Operator_Stop =
;    0490  1  $ASCIC ('!/!/VAX-11 CRD MENU TEST STOPPED BY OPERATOR'),
;    0491  1
;    0492  1 MEN$GT_FncTst_TestStrt_Text =
;    0493  1  $ASCIC ('!/!9AC !9AC !6AC TESTING STARTED - RUN TIME - !5AC MINUTES'),
;    0494  1
;    0495  1 MEN$GT_NULL -
;    0496  1  $ASCIC (''),
;    0497  1
```

```
;    0498  1 %SBTTL ' Warning Messages'
;    0499  1
;    0500  1 MEN$GT_Call_FldSrv =
;    0501  1  $ASCIC ('!/** CALL FIELD SERVICE **'),
;    0502  1
;    0503  1 MEN$GT_InvOperInput =
;    0504  1  $ASCIC ('!/    ** Invalid Choice - Try Again **'),
;    0505  1
;    0506  1 MEN$GT_Failed_Hdwr_Name =
;    0507  1  $ASCIC ('!/** FAILED HARDWARE TESTING: !AC'),
;    0508  1
;    0509  1 MEN$GT_Prep_Error_Text =
; P  0510  1  $ASCIC ('!/!/** !#* !AC !#* **',
;    0511  1        '!/** !#* !AC - INCORRECT HARDWARE TEST PREPARATION !#* **!/'),
;    0512  1
;    0513  1 MEN$GT_Prep_Error_Text_No_User =
;    0514  1  $ASCIC ('!/!/** !#* !AC - INCORRECT HARDWARE TEST PREPARATION !#* **!/'),
;    0515  1
;    0516  1 MEN$GT_Failed_ONL_AUTOSIZER =
;    0517  1  $ASCIC ('!/!/** FAILED TO RUN ONLINE AUTOSIZER  **!/'),
;    0518  1
;    0519  1 !+
;    0520  1 ! Scratch Media Warning
;    0521  1 !-
;    0522  1
;    0523  1 MEN$GT_ScrMedia_Msg =
; P  0524  1  $ASCIC ('!/!/!28** WARNING !29**',
; P  0525  1    '!/* MEDIA on the following hardware MAY BE DESTROYED **',  ![10]
; P  0526  1    '!/*!64* *',
; P  0527  1    '!/*!15* Hardware Name!10* Generic Name!14* *',
; P  0528  1    '!/*!15* ---- --------!10* ------------!14* *'
;    0529  1    ),
;    0530  1
;    0531  1 MEN$GT_ScrMedia_Hdwr =
;    0532  1  $ASCIC ('!/*!15* !22<!AC.> !25<!AC!> *'),
;    0533  1
;    0534  1 MEN$GT_ScrMedia_Hdwr_End =
;    0535  1  $ASCIC ('!/!66**'),
;    0536  1
;    0537  1 MEN$GT_ScrMedia_Prompt =
; P  0538  1  $ASCIC (
; P  0539  1    %CHAR (13,10,13,10),
; P  0540  1    '    Press RETURN when the media is ready to be tested: '
;    0541  1    ),
```

```
;    0542  1 %SBTTL 'Error Messages'
;    0543  1
;    0544  1 MEN$GT_NoAll_NonPgMem =
;    0545  1  $ASCIC ('!/** FAILED TO ALLOCATE MEMORY **'),
;    0546  1
;    0547  1 MEN$GT_Support_File_Not_Found =
;    0548  1  $ASCIC ('!/THE !AC FILE WAS NOT FOUND!! CRD MENU TEST CANNOT CONTINUE PROCESSING'),
;    0549  1
;    0550  1 MEN$GT_Support_File_Load_Err =
;    0551  1  $ASCIC ('!/THE !AC FILE COULD NOT BE LOADED!! CRD MENU TEST CANNOT CONTINUE PROCESSING'),
```

```
;    0552  1 %SBTTL 'Hardware Test Preparation Text'
;    0553  1
;    0554  1 !+
;    0555  1 ! Device Test Preparation text
;    0556  1 !-
;    0557  1
;    0558  1 MEN$GT_DevTstPrep_Name =
;    0559  1  $ASCIC ('!/   Hardware: !AD'),
;    0560  1
;    0561  1 MEN$GT_DevTstPrep_Text =
;    0562  1  $ASCIC ('!/Preparation: '),
;    0563  1
;    0564  1 MEN$GT_DevTstPrep_Null =
;    0565  1  $ASCIC ('1. No Hardware test preparation required.!/'),
;    0566  1
;    0567  1 MEN$GT_DevTstPrep_1 =
; P  0568  1  $ASCIC ('1. Insert disk cartridge in drive (disk will not be corrupted).!/',  ![10]
; P  0569  1    '!14* 2. Push in  RUN STOP  or 'LOAD  switch.!/',
; P  0570  1    '!13* 3. Push out  WRITE PROT  switch (not lighted).!/',
;    0571  1    '!13* 4. Wait for 'READY  indicator to light.'),
;    0572  1
;    0573  1 MEN$GT_DevTstPrep_2 =
; P  0574  1  $ASCIC ('1. Insert floppy diskette in drive and close door.!/',
;    0575  1    '!13*    Current contents of diskette MAY BE DESTROYED.'),   ![10]
;    0576  1
;    0577  1 MEN$GT_DevTstPrep_3 =
; P  0578  1  $ASCIC ('1. Mount tape on drive. Tape must have write-ring.!/',
; P  0579  1    '!13*    Current contents of tape MAY BE DESTROYED.!/',    ![10]
; P  0580  1    '!13* 2. Push in 'LOD  or  LOAD  switch to load tape.!/',
;    0581  1    '!13* 3. Push in 'ONL  or 'ONLINE' switch.'),
;    0582  1
;    0583  1 MEN$GT_DevTstPrep_4 =
; P  0584  1  $ASCIC ('1. Push in  RUN STOP  switch.!/',
; P  0585  1    '!13* 2. Push out  WRITE PROT  switch (not lighted).!/',
;    0586  1    '!13* 3. Wait for  READY  indicator to light.'),
;    0587  1
;    0588  1 MEN$GT_DevTstPrep_5 =
; P  0589  1  $ASCIC ('1. Push in 'LOAD  switch.!/',
; P  0590  1    '!13* 2. Push out  WRITE PROT  switch (not lighted).!/',
;    0591  1    '!13* 3. Wait for  READY  indicator to light.'),
;    0592  1
;    0593  1 MEN$GT_DevTstPrep_6 =
; P  0594  1  $ASCIC ('1. Insert removable disk in drive!/',
; P  0595  1    '!13* 2. Push in  RUN  switch!/',
; P  0596  1    '!13* 3. Push out Write Protect  REMOVE  and  FIXED !/',
; P  0597  1    '!13*    switch (unlighted)!/',
;    0598  1    '!13* 4. Wait for  RUN' light to stop blinking'),
;    0599  1
;    0600  1 MEN$GT_DevTstPrep_7 =
;    0601  1  $ASCIC ('1. Power up drive by pressing (1/0) switch to ON (1)!/');    ![10]
;    0602  1
```

```
;   0603  1 END
;   0604  0 ELUDOM


            .TITLE  CRDTEXT *** CRD MENU - Message Module
            .IDENT  \11\

            .PSECT  WORK,NOEXE,2

            00000 CRD$GA_CRD_MESSAGE_BUFFER::
            .BLKB   132

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      54  58  45  54  44  52  43  07  00000 P.AAA:  .ASCII  <7>\CRDTEXT\          ;
        20  44  4E  41  20  05  00008 P.AAB:  .ASCII  <5>\ AND \          ;
      2F  21  02  0000E P.AAC:  .ASCII  <2>\!/\          ;
      5F  21  02  00011 P.AAD:  .ASCII  <2>\!_\          ;
          2F  01  00014 P.AAE:  .ASCII  <1>\/\          ;
    3F  3F  3F  3F  04  00016 P.AAF:  .ASCII  <4>\????\          ;
    5F  21  5F  21  04  0001B P.AAG:  .ASCII  <4>\!_!_\          ;
    55  4E  45  4D  04  00020 P.AAH:  .ASCII  <4>\MENU\          ;
    4F  54  55  41  04  00025 P.AAI:  .ASCII  <4>\AUTO\          ;
      43  41  38  31  21  2F  21  07  0002A P.AAJ:  .ASCII  <7>\!/!18AC\          ;
54  52  41  54  53  20  47  4E  49  54  53  45  54  20  1D  00032 P.AAK:  .ASCII  <29>\ TESTING STARTED - RUN TIME  \       ;
3D  20  45  4D  49  54  20  4E  55  52  20  2D  20  44  45  00041                ;
    53  45  54  55  4E  49  4D  20  43  41  34  21  20  0D  00050 P.AAL:  .ASCII  <13>\ !4AC MINUTES\          ;
    2F  21  2A  44  45  4C  49  41  46  2A  20  20  20  0D  0005E P.AAM:  .ASCII  <13>\   *FAILED*!/\          ;
    2F  21  20  44  45  53  53  41  50  20  20  20  20  0D  0006C P.AAN:  .ASCII  <13>\    PASSED !/\          ;
43  20  54  53  45  54  20  43  41  21  2F  21  2F  21  21  0007A P.AAO:  .ASCII  \!!/!/!AC TEST COMPLETE - NO ERRORS\       ;
52  45  20  4F  4E  20  2D  20  45  54  45  4C  50  4D  4F  00089                ;
        53  52  4F  52  00098                ;
43  20  54  53  45  54  20  43  41  21  2F  21  2F  21  1E  0009C P.AAP:  .ASCII  <30>\!/!/!AC TEST COMPLETE - ERRORS\       ;
52  4F  52  52  45  20  2D  20  45  54  45  4C  50  4D  4F  000AB                ;
      53  000BA                ;
49  20  54  53  45  54  20  43  41  21  2F  21  2F  21  36  000BB P.AAQ:  .ASCII  \6!/!/!AC TEST INCOMPLETE   !AC BAD - CAL\  ;
43  41  21  20  2D  20  45  54  45  4C  50  4D  4F  43  4E  000CA                ;
    4C  41  43  20  2D  20  44  41  42  20  000D9                ;
45  43  49  56  52  45  53  20  44  4C  45  49  46  20  4C  000E3    .ASCII  \L FIELD SERVICE\          ;
49  20  54  53  45  54  20  43  41  21  2F  21  2F  21  42  000F2 P.AAR:  .ASCII  \B!/!/!AC TEST INCOMPLETE   IF ASSISTANCE\  ;
20  46  49  20  2D  20  45  54  45  4C  50  4D  4F  43  4E  00101                ;
      45  43  4E  41  54  53  49  53  53  41  00110                ;
46  20  4C  4C  41  43  20  2C  44  45  44  45  45  4E  20  0011A    .ASCII  \ NEEDED, CALL FIELD SERVICE\          ;
      45  43  49  56  52  45  53  20  44  4C  45  49  00129                ;
49  20  54  53  45  54  20  43  41  21  2F  21  2F  21  52  00135 P.AAS:  .ASCII  \R!/!/!AC TEST INCOMPLETE /CHECK DEVICE P\  ;
43  45  48  43  2F  21  45  54  45  4C  50  4D  4F  43  4E  00144                ;
      50  20  45  43  49  56  45  44  20  4B  00153                ;
46  49  20  2D  20  4E  4F  49  54  41  52  41  50  45  52  0015D    .ASCII  \REPARATION - IF OK, THEN CALL FIELD SERV\  ;
20  4C  4C  41  43  20  4E  45  48  54  20  2C  4B  4F  20  0016C                ;
      56  52  45  53  20  44  4C  45  49  46  0017B                ;
      45  43  49  00185    .ASCII  \ICE\          ;
41  20  54  53  45  54  20  43  41  21  2F  21  2F  21  29  00188 P.AAT:  .ASCII  \)!/!/.AC TEST ABORTED - CALL FIELD SERVI\  ;
46  20  4C  4C  41  43  20  2D  20  44  45  54  52  4F  42  00197                ;
      49  56  52  45  53  20  44  4C  45  49  001A6                ;
          45  43  001B0    .ASCII  \CE\          ;
```

```
41  20  54  53  45  54  20  43  41  21  2F  21  2F  21  29  001B2 P.AAU:  .ASCII  \)!/!/!AC TEST ABORTED - CALL FIELD SERVI\  ;
46  20  4C  4C  41  43  20  2D  20  44  45  54  52  4F  42  001C1              ;
      49  56  52  45  53  20  44  4C  45  49  001D0                            ;
          45  43  001DA     .ASCII  \CE\                ;
41  20  54  53  45  54  20  43  41  21  2F  21  2F  21  29  001DC P.AAV:  .ASCII  \)!/!/!AC TEST ABORTED - CALL FIELD SERVI\  ;
46  20  4C  4C  41  43  20  2D  20  44  45  54  52  4F  42  001EB              ;
      49  56  52  45  53  20  44  4C  45  49  001FA                            ;
          45  43  00204     .ASCII  \CE\                ;
41  20  54  53  45  54  20  43  41  21  2F  21  2F  21  29  00206 P.AAW:  .ASCII  \)!/!/!AC TEST ABORTED - CALL FIELD SERVI\  ;
46  20  4C  4C  41  43  20  2D  20  44  45  54  52  4F  42  00215              ;
      49  56  52  45  53  20  44  4C  45  49  00224                            ;
          45  43  0022E     .ASCII  \CE\                ;
41  20  54  53  45  54  20  43  41  21  2F  21  2F  21  32  00230 P.AAX:  .ASCII  \2!/!/!AC TEST ABORTED - CONTROL-C TYPED \  ;
4F  52  54  4E  4F  43  20  2D  20  44  45  54  52  4F  42  0023F              ;
      20  44  45  50  59  54  20  43  2D  4C  0024E                            ;
  4C  41  4E  49  4D  52  45  54  20  54  41  00258     .ASCII  \AT TERMINAL\           ;
20  46  4F  20  44  4E  45  20  2A  2A  2A  2A  2F  21  26  00263 P.AAY:  .ASCII  \&!/**** END OF VAX-11 CRD !AC TEST ****\   ;
20  43  41  21  20  44  52  43  20  31  31  2D  58  41  56  00272              ;
    2A  2A  2A  2A  20  54  53  45  54  00281                ;
20  47  4E  49  4E  52  55  54  45  52  2F  21  2F  21  3C  0028A P.AAZ:  .ASCII  \<!/!/RETURNING TO VAX-11 CRD CONSOLE, WA\  ;
43  20  44  52  43  20  31  31  2D  58  41  56  20  4F  54  00299              ;
      41  57  20  2C  45  4C  4F  53  4E  4F  002A8                            ;
52  50  20  22  3E  3E  3E  22  20  52  4F  46  20  54  49  002B2    .ASCII  \IT FOR  >>>  PROMPT!/\        ;
          2F  21  54  50  4D  4F  002C1            ;
20  47  4E  49  4E  52  55  54  45  52  2F  21  2F  21  39  002C7 P.ABA:  .ASCII  \9!/!/RETURNING TO VAX DIAGNOSTIC SUPERVI\  ;
54  53  4F  4E  47  41  49  44  20  58  41  56  20  4F  54  002D6              ;
      49  56  52  45  50  55  53  20  43  49  002E5                            ;
44  4F  4D  20  44  4E  41  4D  4D  4F  43  20  52  4F  53  002EF    .ASCII  \SOR COMMAND MODE!/\         ;
      2F  21  45  002FE                ;
66  20  4E  52  55  54  45  52  20  73  73  65  72  50  18  00301 P.ABB:  .ASCII  <24>\Press RETURN for more...\       ;
      2E  2E  2E  65  72  6F  6D  20  72  6F  00310                            ;
66  20  4E  52  55  54  45  52  20  73  73  65  72  50  1D  0031A P.ABC:  .ASCII  <29>\Press RETURN for MAIN MENU...\       ;
2E  2E  2E  55  4E  45  4D  20  4E  49  41  4D  20  72  6F  00329                ;
44  45  54  52  4F  50  50  55  53  20  4F  4E  2F  21  36  00338 P.ABD:  .ASCII  \6!/NO SUPPORTED DISK CONTROLLER FOUND BY\  ;
45  4C  4C  4F  52  54  4E  4F  43  20  4B  53  49  44  20  00347              ;
      59  42  20  44  4E  55  4F  46  20  52  00356                            ;
54  53  45  54  20  4F  52  43  41  4D  20  44  52  43  20  00360    .ASCII  \ CRD MACRO TEST\         ;
45  44  20  4F  54  20  45  4C  42  41  4E  55  2F  21  4B  0036F P.ABE:  .ASCII  \K!/UNABLE TO DETERMINE DEVICES ON THE SY\  ;
53  45  43  49  56  45  44  20  45  4E  49  4D  52  45  54  0037E              ;
      59  53  20  45  48  54  20  4E  4F  20  0038D                            ;
48  54  52  55  46  20  4F  4E  20  2D  20  4D  45  54  53  00397    .ASCII  \STEM - NO FURTHER TESTING POSSIBLE!/\       ;
53  53  4F  50  20  47  4E  49  54  53  45  54  20  52  45  003A6              ;
          2F  21  45  4C  42  49  003B5                ;
41  20  54  4F  4E  20  45  43  49  56  45  44  2F  21  18  003BB P.ABF:  .ASCII  <24>\!/DEVICE NOT AVAILABLE!/\        ;
      2F  21  45  4C  42  41  4C  49  41  56  003CA                            ;
41  56  41  20  54  4F  4E  20  45  43  49  56  45  44  16  003D4 P.ABG:  .ASCII  <22>\DEVICE NOT AVAILABLE!/\         ;
      2F  21  45  4C  42  41  4C  49  003E3                ;
4E  47  41  49  44  20  45  43  49  56  45  44  2F  21  1F  003EB P.ABH:  .ASCII  <31>\!/DEVICE DIAGNOSTIC NOT FOUND!/\        ;
44  4E  55  4F  46  20  54  4F  4E  20  43  49  54  53  4F  003FA              ;
          2F  21  00409                ;
49  54  4E  4F  43  20  54  4F  4E  4E  41  43  2F  21  27  0040B P.ABI:  .ASCII  \'!/CANNOT CONTINUE TESTING THIS DEVICE!/\  ;
49  48  54  20  47  4E  49  54  53  45  54  20  45  55  4E  0041A              ;
      2F  21  45  43  49  56  45  44  20  53  00429                            ;
      43  41  21  20  2A  2A  2A  2F  21  09  00433 P.ABJ:  .ASCII  <9>\!/*** !AC\          ;
49  53  52  45  56  20  53  49  48  54  2F  21  2F  21  2F  0043D P.ABK:  .ASCII  \/!/!/THIS VERSION OF CRD !AC TEST IS COR\  ;
```

```
54 20 43 41 21 20 44 52 43 20 46 4F 20 4E 4F 0044C                ;
      52 4F 43 20 53 49 20 54 53 45 0045B                         ;
        2F 21 44 45 54 50 55 52 00465     .ASCII  \RUPTED!/\              ;
61 6D 6D 6F 43 20 53 44 20 2A 2A 2A 2F 21 1D 0046D P.ABL:  .ASCII  <29>\!/*** DS Command: !60<!AD!>!/\       ;
2F 21 3E 21 44 41 21 3C 30 36 21 20 3A 64 6E 0047C                ;
        00 0048B P.ABM:  .ASCII  <0>                ;
      2F 21 02 0048C P.ABN:  .ASCII  <2>\!/\              ;
        20 2A 2A 2F 21 05 0048F P.ABO:  .ASCII  <5>\!/** \          ;
          2A 2A 20 03 00495 P.ABP:  .ASCII  <3>\ **\              ;
59 52 41 4D 4D 55 53 20 2D 2D 2F 21 2F 21 13 00499 P.ABQ:  .ASCII  <19>\!/!/-- SUMMARY --!/\        ;
      2F 21 2D 2D 20 004A8                  ;
20 3A 64 65 6C 69 61 46 20 20 20 20 2F 21 2D 004AD P.ABR:  .ASCII  \-!/    Failed:  (Hardware name/Generic n\  ;
65 6D 61 6E 20 65 72 61 77 64 72 61 48 28 20 004BC                ;
      6E 20 63 69 72 65 6E 65 47 2F 004CB                         ;
        2F 21 29 65 6D 61 004D5     .ASCII  \ame)!/\              ;
20 3A 64 65 73 73 61 50 20 20 20 20 2F 21 2D 004DB P.ABS:  .ASCII  \-!/    Passed:  (Hardware name/Generic n\  ;
65 6D 61 6E 20 65 72 61 77 64 72 61 48 28 20 004EA                ;
      6E 20 63 69 72 65 6E 65 47 2F 004F9                         ;
        2F 21 29 65 6D 61 00503     .ASCII  \ame)!/\              ;
63 65 72 72 6F 63 6E 49 20 20 20 20 2F 21 45 00509 P.ABT:  .ASCII  \E!/    Incorrect Hardware Preparation:  \  ;
70 65 72 50 20 65 72 61 77 64 72 61 48 20 74 00518                ;
      20 20 3A 6E 6F 69 74 61 72 61 00527                         ;
2F 65 6D 61 6E 20 65 72 61 77 64 72 61 48 28 00531     .ASCII  \(Hardware name/Generic name)!/\       ;
2F 21 29 65 6D 61 6E 20 63 69 72 65 6E 65 47 00540                ;
20 3A 73 72 65 68 74 4F 20 20 20 20 2F 21 25 0054F P.ABU:  .ASCII  \%!/    Others:  (Diagnostic message)!/\    ;
65 6D 20 63 69 74 73 6F 6E 67 61 69 44 28 20 0055E                ;
        2F 21 29 65 67 61 73 73 0056D                   ;
4D 45 4D 5F 21 5F 21 5F 21 55 50 43 5F 21 15 00575 P.ABV:  .ASCII  <21>\!_CPU!_!_!_MEMORY!_! \        ;
      5F 21 5F 21 59 52 4F 00584                  ;
4F 54 53 55 43 20 2A 39 31 21 2F 21 2F 21 D1 0058B P.ABW:  .ASCII  <209>\!/!/!19* CUSTOMER RUNNABLE DIAGNOS\   ;
49 44 20 45 4C 42 41 4E 4E 55 52 20 52 45 4D 0059A                ;
        53 4F 4E 47 41 005A9                    ;
2F 21 2F 21 45 47 41 4B 43 41 50 20 43 49 54 005AE     .ASCII  \TIC PACKAGE!/!/!30* Version   V020!/!/!/\  ;
20 20 20 6E 6F 69 73 72 65 56 20 2A 30 33 21 005BD                ;
      2F 21 2F 21 2F 21 30 32 30 56 005CC                         ;
2D 58 41 56 20 4E 49 47 45 42 20 2A 2A 34 21 005D6     .ASCII  \!4** BEGIN VAX-11 CRD MENU TEST !4**!/!/\  ;
53 45 54 20 55 4E 45 4D 20 44 52 43 20 31 31 005E5                ;
      2F 21 2F 21 2A 2A 34 21 20 54 005F4                         ;
6B 20 4C 52 54 43 20 65 68 74 20 65 70 79 54 005FE     .ASCII  \Type the CTRL key and the C key (togethe\  ;
65 6B 20 43 20 65 68 74 20 64 6E 61 20 79 65 0060D                ;
        65 68 74 65 67 6F 74 28 20 79 0061C                     ;
65 6D 69 74 20 79 6E 61 2F 21 74 61 20 29 72 00626     .ASCII  \r) at!/any time to interrupt Menu Test p\  ;
4D 20 74 70 75 72 72 65 74 6E 69 20 6F 74 20 00635                ;
        70 20 74 73 65 54 20 75 6E 65 00644                     ;
2F 21 2F 21 2F 21 67 6E 69 73 73 65 63 6F 72 0064E     .ASCII  \rocessing!/!/!/\           ;
49 20 45 52 41 57 44 52 41 48 2D 58 41 56 4F 0065D P.ABX:  .ASCII  \OVAX-HARDWARE IDENTIFICATION - RUN TIME \  ;
2D 20 4E 4F 49 54 41 43 49 46 49 54 4E 45 44 0066C                ;
        20 45 4D 49 54 20 4E 55 52 20 0067B                     ;
53 45 54 55 4E 49 4D 20 30 33 3A 30 30 20 3D 00685     .ASCII  \= 00:30 MINUTES!/!/Please wait . . .!/!/\  ;
74 69 61 77 20 65 73 61 65 6C 50 2F 21 2F 21 00694                ;
        2F 21 2F 21 2E 20 2E 20 2E 20 006A3                     ;
52 41 48 20 44 52 43 20 31 31 2D 58 41 56 2D 006AD P.ABY:  .ASCII  \-VAX-11 CRD HARDWARE IDENTIFICATION COMP\  ;
43 49 46 49 54 4E 45 44 49 20 45 52 41 57 44 006BC                ;
      50 4D 4F 43 20 4E 4F 49 54 41 006CB                         ;
        2F 21 45 54 45 4C 006D5     .ASCII  \LETE!/\              ;
```

```
49 41 4D 2F 21 2D 2A 37 32 21 2F 21 2F 21 2D  006DB P.ABZ:  .ASCII  \-!/!/!27*-!/MAIN MENU - Functional Test!\  ;
69 74 63 6E 75 46 20 2D 20 55 4E 45 4D 20 4E  006EA                ;
   21 74 73 65 54 20 6C 61 6E 6F              006F9                ;
      2D 2A 37 32 21 2F                       00703    .ASCII  \/!27*-\          ;
41 21 20 3D 20 44 41 21 20 20 20 20 2F 21 0F  00709 P.ACA:  .ASCII  <15>\!/    !AD = !AC\          ;
   43                                         00718                ;
74 73 65 54 20 75 6E 65 4D 20 74 69 78 45 0E  00719 P.ACB:  .ASCII  <14>\Exit Menu Test\         ;
74 61 72 61 70 65 72 70 20 74 6E 69 72 50 39  00728 P.ACC:  .ASCII  \9Print preparation required for test of \  ;
6F 66 20 64 65 72 69 75 71 65 72 20 6E 6F 69  00737                ;
   20 66 6F 20 74 73 65 74 20 72              00746                ;
77 64 72 61 68 20 64 65 74 72 6F 70 70 75 73  00750    .ASCII  \supported hardware\         ;
   65 72 61                                   0075F                ;
73 65 54 20 64 6E 61 20 74 63 65 6C 65 53 27  00762 P.ACD:  .ASCII  \'Select and Test hardware from TEST MENU\  ;
6D 6F 72 66 20 65 72 61 77 64 72 61 68 20 74  00771                ;
   55 4E 45 4D 20 54 53 45 54 20              00780                ;
20 66 6F 20 74 73 69 6C 20 74 6E 69 72 50 3B  0078A P.ACE:  .ASCII  \;Print list of identified system hardwar\  ;
74 73 79 73 20 64 65 69 66 69 74 6E 65 64 69  00799                ;
   72 61 77 64 72 61 68 20 6D 65              007A8                ;
73 20 74 72 6F 70 70 75 73 20 64 6E 61 20 65  007B2    .ASCII  \e and support status\       ;
   73 75 74 61 74                             007C1                ;
6F 70 70 75 53 20 4C 4C 41 20 54 53 45 54 1B  007C6 P.ACF:  .ASCII  <27>\TEST ALL Supported Hardware\      ;
65 72 61 77 64 72 61 48 20 64 65 74 72        007D5                ;
6F 20 65 70 79 54 20 20 20 20 2F 21 2F 21 44  007E2 P.ACG:  .ASCII  \D!/!/    Type one of the above (for exam\  ;
65 76 6F 62 61 20 65 68 74 20 66 6F 20 65 6E  007F1                ;
   6D 61 78 65 20 72 6F 66 28 20              00800                ;
72 70 20 64 6E 61 20 2C 29 41 20 2C 65 6C 70  0080A    .ASCII  \ple, A), and press RETURN!/!/\      ;
   2F 21 2F 21 4E 52 55 54 45 52 20 73 73 65  00819                ;
4E 49 41 4D 20 72 65 74 6E 45 20 20 20 20 1C  00827 P.ACH:  .ASCII  <28>\    Enter MAIN MENU choice: \      ;
   20 3A 65 63 69 6F 68 63 20 55 4E 45 4D 20  00836                ;
54 53 45 54 2F 21 2D 2A 39 21 2F 21 2F 21 19  00844 P.ACI:  .ASCII  <25>\!/!/!9*-!/TEST MENU!/!9*-\      ;
   2D 2A 39 21 2F 21 55 4E 45 4D 20           00853                ;
21 4C 55 21 44 41 21 3C 33 21 20 20 20 20 20  0085E P.ACJ:  .ASCII  \     !3<!AD!UL!> = !9AC !9AC !6AC\      ;
21 20 43 41 39 21 20 43 41 39 21 20 3D 20 3E  0086D                ;
   43 41 36 20                                0087C                ;
21 4C 55 21 44 41 21 3C 33 21 20 20 20 20 3A  0087F P.ACK:  .ASCII  \:    !3<!AD!UL!> = TEST ALL of the above\  ;
66 6F 20 4C 4C 41 20 54 53 45 54 20 3D 20 3E  0088E                ;
   65 76 6F 62 61 20 65 68 74 20              0089D                ;
64 72 61 68 20 64 65 74 72 6F 70 70 75 73 20  008A7    .ASCII  \ supported hardware\        ;
   65 72 61 77                                008B6                ;
6F 20 65 70 79 54 20 20 20 20 2F 21 2F 21 56  008BA P.ACL:  .ASCII  \V!/!/    Type one of the above (for exam\  ;
65 76 6F 62 61 20 65 68 74 20 66 6F 20 65 6E  008C9                ;
   6D 61 78 65 20 72 6F 66 28 20              008D8                ;
70 20 64 6E 61 20 2C 29 31 41 20 2C 65 6C 70  008E2    .ASCII  \ple, A1), and press RETURN to start test\  ;
20 6F 74 20 4E 52 55 54 45 52 20 73 73 65 72  008F1                ;
   74 73 65 74 20 74 72 61 74 73              00900                ;
   2F 21 2F 21 67 6E 69                       0090A    .ASCII  \ing!/!/\           ;
54 53 45 54 20 72 65 74 6E 45 20 20 20 20 1C  00911 P.ACM:  .ASCII  <28>\    Enter TEST MENU choice: \      ;
   20 3A 65 63 69 6F 68 63 20 55 4E 45 4D 20  00920                ;
55 54 45 52 20 73 73 65 72 50 20 2A 2A 2A 37  0092E P.ACN:  .ASCII  \7*** Press RETURN for remainder of TEST \  ;
65 64 6E 69 61 6D 65 72 20 72 6F 66 20 4E 52  0093D                ;
   20 54 53 45 54 20 66 6F 20 72              0094C                ;
2A 2A 20 73 65 63 69 6F 68 63 20 55 4E 45 4D  00956    .ASCII  \MENU choices ***\           ;
   2A                                         00965                ;
66 20 4E 52 55 54 45 52 20 73 73 65 72 50 1D  00966 P.ACO:  .ASCII  <29>\Press RETURN for TEST MENU...\      ;
2E 2E 2E 55 4E 45 4D 20 54 53 45 54 20 72 6F  00975                ;
```

```
20 2F 21 2D 2A 34 31 21 20 20 20 20 2F 21 2C  00984 P.ACP:  .ASCII  \,!/    !14*-!/    CONTROL-C MENU!/    !1\  ;
45 4D 20 43 2D 4C 4F 52 54 4E 4F 43 20 20 20  00993              ;
      31 21 20 20 20 20 2F 21 55 4E  009A2              ;
   2F 21 2D 2A 34  009AC    .ASCII  \4*-!/\           ;
20 44 41 21 20 20 20 20 20 20 20 20 2F 21 13  009B1 P.ACQ:  .ASCII  <19>\!/        !AD = !AC\        ;
   43 41 21 20 3D  009C0           ;
73 73 65 63 6F 72 70 20 65 6D 75 73 65 52 2B  009C5 P.ACR:  .ASCII  \+Resume process interrupted by the CONTR\  ;
79 62 20 64 65 74 70 75 72 72 65 74 6E 69 20  009D4              ;
   52 54 4E 4F 43 20 65 68 74 20  009E3              ;
      43 2D 4C 4F  009ED    .ASCII  \OL-C\          ;
72 72 75 63 20 65 68 74 20 74 72 6F 62 41 36  009F1 P.ACS:  .ASCII  \6Abort the current process, and return t\  ;
6E 61 20 2C 73 73 65 63 6F 72 70 20 74 6E 65  00A00              ;
   74 20 6E 72 75 74 65 72 20 64  00A0F              ;
55 4E 45 4D 20 4E 49 41 4D 20 65 68 74 20 6F  00A19    .ASCII  \o the MAIN MENU\        ;
79 54 20 20 20 20 20 20 20 20 2F 21 2F 21 47  00A28 P.ACT:  .ASCII  \G!/!/        Type one of the above (for \  ;
61 20 65 68 74 20 66 6F 20 65 6E 6F 20 65 70  00A37              ;
   20 72 6F 66 28 20 65 76 6F 62  00A46              ;
64 6E 61 20 29 41 20 2C 65 6C 70 6D 61 78 65  00A50    .ASCII  \example, A) and press RETURN!/!/\       ;
2F 21 4E 52 55 54 45 52 20 73 73 65 72 70 20  00A5F              ;
      2F 21  00A6E           ;
20 72 65 74 6E 45 20 20 20 20 20 20 20 20 25  00A70 P.ACU:  .ASCII  \%        Enter CONTROL-C MENU choice: \    ;
20 55 4E 45 4D 20 43 2D 4C 4F 52 54 4E 4F 43  00A7F              ;
   20 3A 65 63 69 6F 68 63  00A8E           ;
65 68 74 20 67 6E 69 6D 75 73 65 52 2F 21 24  00A96 P.ACV:  .ASCII  \$!/Resuming the interrupted process!/\    ;
72 70 20 64 65 74 70 75 72 72 65 74 6E 69 20  00AA5              ;
   2F 21 73 73 65 63 6F  00AB4           ;
      20 20 20 20 20 20 55 50 43 09  00ABB P.ACW:  .ASCII  <9>\CPU      \          ;
      20 20 20 59 52 4F 4D 45 4D 09  00AC5 P.ACX:  .ASCII  <9>\MEMORY   \          ;
      20 20 20 20 20 4B 53 49 44 09  00ACF P.ACY:  .ASCII  <9>\DISK     \          ;
      20 20 20 20 20 45 50 41 54 09  00AD9 P.ACZ:  .ASCII  <9>\TAPE     \          ;
      20 20 20 20 2E 4D 4D 4F 43 09  00AE3 P.ADA:  .ASCII  <9>\COMM.    \          ;
      20 45 4D 49 54 4C 41 45 52 09  00AED P.ADB:  .ASCII  <9>\REALTIME \          ;
      20 4C 41 4E 49 4D 52 45 54 09  00AF7 P.ADC:  .ASCII  <9>\TERMINAL \          ;
      20 20 52 45 54 4E 49 52 50 09  00B01 P.ADD:  .ASCII  <9>\PRINTER  \          ;
      20 20 20 20 20 44 52 41 43 09  00B0B P.ADE:  .ASCII  <9>\CARD     \          ;
      20 20 52 4F 54 50 41 44 41 09  00B15 P.ADF:  .ASCII  <9>\ADAPTOR  \          ;
      20 20 20 20 20 20 20 20 20 09  00B1F P.ADG:  .ASCII  <9>\         \          ;
77 64 72 61 48 3C 39 21 20 2A 38 21 2F 21 6B  00B29 P.ADH:  .ASCII  \k!/!8* !9<Hardware!> !8<Generic!> !9<Tes\  ;
69 72 65 6E 65 47 3C 38 21 20 3E 21 65 72 61  00B38              ;
   73 65 54 3C 39 21 20 3E 21 63  00B47              ;
21 20 2A 38 21 2F 21 3E 21 75 6E 65 4D 20 74  00B51    .ASCII  \t Menu!>!/!8* !9<Name!> !8<Name!> !9<Sup\  ;
6D 61 4E 3C 38 21 20 3E 21 65 6D 61 4E 3C 39  00B60              ;
   70 75 53 3C 39 21 20 3E 21 65  00B6F              ;
2A 39 21 20 2A 38 21 2F 21 3E 21 74 72 6F 70  00B79    .ASCII  \port!>!/!8* !9*- !8*- !9*-!/\       ;
 2F 21 2D 2A 39 21 20 2D 2A 38 21 20 2D  00B88           ;
20 43 41 38 21 20 43 41 39 21 20 2A 38 21 11  00B95 P.ADI:  .ASCII  <17>\!8* !9AC !8AC !AC\        ;
      43 41 21  00BA4           ;
      44 45 54 52 4F 50 50 55 53 09  00BA7 P.ADJ:  .ASCII  <9>\SUPPORTED\         ;
   74 72 6F 70 70 75 53 20 6F 4E 0A  00BB1 P.ADK:  .ASCII  <10>\No Support\        ;
20 66 6F 20 64 6E 45 5B 5F 21 5F 21 2F 21 13  00BBC P.ADL:  .ASCII  <19>\!/!_!_[End of list]\       ;
   5D 74 73 69 6C  00BCB           ;
74 20 65 70 79 54 20 2A 34 21 2F 21 2F 21 63  00BD0 P.ADM:  .ASCII  \c!/!/!4* Type the CTRL key and the C key\  ;
64 6E 61 20 79 65 6B 20 4C 52 54 43 20 65 68  00BDF              ;
   79 65 6B 20 43 20 65 68 74 20  00BEE              ;
34 21 2F 21 29 72 65 68 74 65 67 6F 74 28 20  00BF8    .ASCII  \ (together)!/!4* at any time to interrup\  ;
```

```
74 20 65 6D 69 74 20 79 6E 61 20 74 61 20 2A 00C07                  ;
   70 75 72 72 65 74 6E 69 20 6F 00C16                              ;
65 54 20 6C 61 6E 6F 69 74 63 6E 75 46 20 74 00C20    .ASCII  \t Functional Testing\        ;
   67 6E 69 74 73 00C2F                  ;
2A 30 31 21 20 2A 34 21 2F 21 2F 21 2F 21 2E 00C34 P.ADN:  .ASCII  \.!/!/!/!4* !10** BEGIN FUNCTIONAL TESTIN\  ;
4F 49 54 43 4E 55 46 20 4E 49 47 45 42 20 2A 00C43                  ;
   4E 49 54 53 45 54 20 4C 41 4E 00C52                              ;
   2A 2A 30 31 21 20 47 00C5C    .ASCII  \G !10**\                  ;
54 20 4E 55 52 20 2A 35 31 21 2F 21 2F 21 29 00C63 P.ADO:  .ASCII  \)!/!/!15* RUN TIME = !2ZL:!2ZL MINUTES!/\  ;
4C 5A 32 21 3A 4C 5A 32 21 20 3D 20 45 4D 49 00C72                  ;
   2F 21 53 45 54 55 4E 49 4D 20 00C81                              ;
      2F 21 00C8B    .ASCII  \!/\                                   ;
54 20 4C 41 4E 4F 49 54 43 4E 55 46 2F 21 29 00C8D P.ADP:  .ASCII  \)!/FUNCTIONAL TESTING COMPLETE - NO ERRO\  ;
45 54 45 4C 50 4D 4F 43 20 47 4E 49 54 53 45 00C9C                  ;
   4F 52 52 45 20 4F 4E 20 2D 20 00CAB                              ;
      53 52 00CB5    .ASCII  \RS\                                   ;
41 4E 4F 49 54 43 4E 55 46 20 2A 2A 2F 21 2B 00CB7 P.ADQ:  .ASCII  \+!/** FUNCTIONAL TESTING INCOMPLETE - ER\  ;
4D 4F 43 4E 49 20 47 4E 49 54 53 45 54 20 4C 00CC6                  ;
   52 45 20 2D 20 45 54 45 4C 50 00CD5                              ;
      53 52 4F 52 00CDF    .ASCII  \RORS\                 ;
41 4E 4F 49 54 43 4E 55 46 20 2A 2A 2F 21 29 00CE3 P.ADR:  .ASCII  \)!/** FUNCTIONAL TESTING COMPLETE - ERRO\  ;
4C 50 4D 4F 43 20 47 4E 49 54 53 45 54 20 4C 00CF2                  ;
   4F 52 52 45 20 2D 20 45 54 45 00D01                              ;
      53 52 00D0B    .ASCII  \RS\                                   ;
41 4E 4F 49 54 43 4E 55 46 20 2A 2A 2F 21 22 00D0D P.ADS:  .ASCII  \ !/** FUNCTIONAL TESTING INCOMPLETE\       ;
4D 4F 43 4E 49 20 47 4E 49 54 53 45 54 20 4C 00D1C                  ;
   45 54 45 4C 50 00D2B                  ;
54 53 49 53 53 41 20 46 49 20 2A 2A 2F 21 31 00D30 P.ADT:  .ASCII  \1!/** IF ASSISTANCE IS NEEDED - CALL FIE\  ;
20 44 45 44 45 45 4E 20 53 49 20 45 43 4E 41 00D3F                  ;
   45 49 46 20 4C 4C 41 43 20 2D 00D4E                              ;
   45 43 49 56 52 45 53 20 44 4C 00D58    .ASCII  \LD SERVICE\           ;
44 52 43 20 31 31 2D 58 41 56 2F 21 2F 21 20 00D62 P.ADU:  .ASCII  \ !/!/VAX-11 CRD MENU TEST ABORTED\       ;
52 4F 42 41 20 54 53 45 54 20 55 4E 45 4D 20 00D71                  ;
   44 45 54 00D80                  ;
44 52 43 20 31 31 2D 58 41 56 2F 21 2F 21 2C 00D83 P.ADV:  .ASCII  \,!/!/VAX-11 CRD MENU TEST ABORTED BY OPE\  ;
52 4F 42 41 20 54 53 45 54 20 55 4E 45 4D 20 00D92                  ;
   45 50 4F 20 59 42 20 44 45 54 00DA1                              ;
   52 4F 54 41 52 00DAB    .ASCII  \RATOR\                ;
44 52 43 20 31 31 2D 58 41 56 2F 21 2F 21 2C 00DB0 P.ADW:  .ASCII  \,!/!/VAX-11 CRD MENU TEST STOPPED BY OPE\  ;
50 4F 54 53 20 54 53 45 54 20 55 4E 45 4D 20 00DBF                  ;
   45 50 4F 20 59 42 20 44 45 50 00DCE                              ;
   52 4F 54 41 52 00DD8    .ASCII  \RATOR\                ;
36 21 20 43 41 39 21 20 43 41 39 21 2F 21 3A 00DDD P.ADX:  .ASCII  \:!/!9AC !9AC !6AC TESTING STARTED - RUN \  ;
52 41 54 53 20 47 4E 49 54 53 45 54 20 43 41 00DEC                  ;
   20 4E 55 52 20 2D 20 44 45 54 00DFB                              ;
4E 49 4D 20 43 41 35 21 20 3D 20 45 4D 49 54 00E05    .ASCII  \TIME = !5AC MINUTES\          ;
      53 45 54 55 00E14                  ;
   00 00E18 P.ADY:  .ASCII  <0>                ;
4C 45 49 46 20 4C 4C 41 43 20 2A 2A 2F 21 1A 00E19 P.ADZ:  .ASCII  <26>\!/** CALL FIELD SERVICE **\        ;
   2A 2A 20 45 43 49 56 52 45 53 20 44 00E28                        ;
6C 61 76 6E 49 20 2A 2A 20 20 20 20 2F 21 26 00E34 P.AEA:  .ASCII  \&!/    ** Invalid Choice - Try Again **\   ;
79 72 54 20 2D 20 65 63 69 6F 68 43 20 64 69 00E43                  ;
   2A 2A 20 6E 69 61 67 41 20 00E52                                 ;
41 48 20 44 45 4C 49 41 46 20 2A 2A 2F 21 21 00E5B P.AEB:  .ASCII  \!!/** FAILED HARDWARE TESTING: !AC\        ;
3A 47 4E 49 54 53 45 54 20 45 52 41 57 44 52 00E6A                  ;
```

```
                43  41  21  20  00E79                ;
43  41  21  20  2A  23  21  20  2A  2A  2F  21  2F  21  50  00E7D P.AEC:  .ASCII  \P!/!/** !#* !AC !#* **!/** !#* !AC - INC\  ;
2A  23  21  20  2A  2A  2F  21  2A  2A  20  2A  23  21  20  00E8C              ;
    43  4E  49  20  2D  20  43  41  21  20  00E9B                  ;
45  52  41  57  44  52  41  48  20  54  43  45  52  52  4F  00EA5    .ASCII  \ORRECT HARDWARE TEST PREPARATION !#* **!\  ;
49  54  41  52  41  50  45  52  50  20  54  53  45  54  20  00EB4              ;
      21  2A  2A  20  2A  23  21  20  4E  4F  00EC3              ;
        2F  00ECD    .ASCII  \/\             ;
43  41  21  20  2A  23  21  20  2A  2A  2F  21  2F  21  3D  00ECE P.AED:  .ASCII  \=!/!/** !#* !AC - INCORRECT HARDWARE TES\  ;
41  48  20  54  43  45  52  52  4F  43  4E  49  20  2D  20  00EDD              ;
      53  45  54  20  45  52  41  57  44  52  00EEC                ;
21  20  4E  4F  49  54  41  52  41  50  45  52  50  20  54  00EF6    .ASCII  \T PREPARATION !#* **!/\        ;
    2F  21  2A  2A  20  2A  23  00F05              ;
20  44  45  4C  49  41  46  20  2A  2A  2F  21  2F  21  2B  00F0C P.AEE:  .ASCII  \+!/!/** FAILED TO RUN ONLINE AUTOSIZER  \  ;
41  20  45  4E  49  4C  4E  4F  20  4E  55  52  20  4F  54  00F1B              ;
      20  20  52  45  5A  49  53  4F  54  55  00F2A                ;
          2F  21  2A  2A  00F34    .ASCII  \**!/\              ;
4E  52  41  57  20  2A  2A  38  32  21  2F  21  2F  21  AE  00F38 P.AEF:  .ASCII  <174>\!/!/!28** WARNING !29**!/* MEDIA o\   ;
45  4D  20  2A  2F  21  2A  2A  39  32  21  20  47  4E  49  00F47              ;
    6F  20  41  49  44  00F56              ;
67  6E  69  77  6F  6C  6C  6F  66  20  65  68  74  20  6E  00F5B    .ASCII  \n the following hardware MAY BE DESTROYE\  ;
42  20  59  41  4D  20  65  72  61  77  64  72  61  68  20  00F6A              ;
      45  59  4F  52  54  53  45  44  20  45  00F79                ;
2A  2F  21  2A  20  2A  34  36  21  2A  2F  21  2A  20  44  00F83    .ASCII  \D *!/*!64* *!/*!15* Hardware Name!10* Ge\  ;
4E  20  65  72  61  77  64  72  61  48  20  2A  35  31  21  00F92              ;
      65  47  20  2A  30  31  21  65  6D  61  00FA1                ;
20  2A  34  31  21  65  6D  61  4E  20  63  69  72  65  6E  00FAB    .ASCII  \neric Name!14* *!/*!15* ------------!10\  ;
2D  2D  2D  2D  2D  2D  20  2A  35  31  21  2A  2F  21  2A  00FBA              ;
      30  31  21  2D  2D  2D  2D  2D  2D  2D  2D  00FC9              ;
21  2D  2D  2D  2D  2D  2D  2D  2D  2D  2D  2D  2D  20  2A  00FD3    .ASCII  \* -----------!14* *\        ;
    2A  20  2A  34  31  00FE2              ;
41  21  3C  32  32  21  20  2A  35  31  21  2A  2F  21  1D  00FE7 P.AEG:  .ASCII  <29>\!/*!15* !22<!AC!> !25<!AC!> *\       ;
2A  20  3E  21  43  41  21  3C  35  32  21  20  3E  21  43  00FF6              ;
        2A  2A  36  36  21  2F  21  07  01005 P.AEH:  .ASCII  <7>\!/!66**\           ;
20  73  73  65  72  50  20  20  20  20  0A  0D  0A  0D  3B  0100D P.AEI:  .ASCII  \;\<13><10><13><10>\    Press RETU\       ;
          55  54  45  52  0101C              ;
64  65  6D  20  65  68  74  20  6E  65  68  77  20  4E  52  01020    .ASCII  \RN when the media is ready to be tested:\  ;
20  6F  74  20  79  64  61  65  72  20  73  69  20  61  69  0102F              ;
      3A  64  65  74  73  65  74  20  65  62  0103E                ;
        20  01048    .ASCII  \ \             ;
4F  54  20  44  45  4C  49  41  46  20  2A  2A  2F  21  21  01049 P.AEJ:  .ASCII  \!!/** FAILED TO ALLOCATE MEMORY **\       ;
52  4F  4D  45  4D  20  45  54  41  43  4F  4C  4C  41  20  01058              ;
          2A  2A  20  59  01067              ;
45  4C  49  46  20  43  41  21  20  45  48  54  2F  21  47  0106B P.AEK:  .ASCII  \G!/THE !AC FILE WAS NOT FOUND!! CRD MENU\  ;
21  44  4E  55  4F  46  20  54  4F  4E  20  53  41  57  20  0107A              ;
      55  4E  45  4D  20  44  52  43  20  21  01089                ;
4F  43  20  54  4F  4E  4E  41  43  20  54  53  45  54  20  01093    .ASCII  \ TEST CANNOT CONTINUE PROCESSING\       ;
49  53  53  45  43  4F  52  50  20  45  55  4E  49  54  4E  010A2              ;
          47  4E  010B1              ;
45  4C  49  46  20  43  41  21  20  45  48  54  2F  21  4D  010B3 P.AEL:  .ASCII  \M!/THE !AC FILE COULD NOT BE LOADED!! CR\  ;
4C  20  45  42  20  54  4F  4E  20  44  4C  55  4F  43  20  010C2              ;
      52  43  20  21  21  44  45  44  41  4F  010D1                ;
4E  41  43  20  54  53  45  54  20  55  4E  45  4D  20  44  010DB    .ASCII  \D MENU TEST CANNOT CONTINUE PROCESSING\    ;
52  50  20  45  55  4E  49  54  4E  4F  43  20  54  4F  4E  010EA              ;
        47  4E  49  53  53  45  43  4F  010F9                ;
```

```
3A 65 72 61 77 64 72 61 48 20 20 20 2F 21 12 01101 P.AEM:  .ASCII  <18>\!/   Hardware: !AD\        ;
        44 41 21 20 01110                          ;
3A 6E 6F 69 74 61 72 61 70 65 72 50 2F 21 0F 01114 P.AEN:  .ASCII  <15>\!/Preparation: \         ;
     20 01123                    ;
65 72 61 77 64 72 61 48 20 6F 4E 20 2E 31 2B 01124 P.AEO:  .ASCII  \+1. No Hardware test preparation require\  ;
69 74 61 72 61 70 65 72 70 20 74 73 65 74 20 01133                 ;
   65 72 69 75 71 65 72 20 6E 6F 01142                             ;
      2F 21 2E 64 0114C    .ASCII  \d.!/\                  ;
6B 73 69 64 20 74 72 65 73 6E 49 20 2E 31 D0 01150 P.AEP:  .ASCII  <208>\1. Insert disk cartridge in drive \   ;
64 20 6E 69 20 65 67 64 69 72 74 72 61 63 20 0115F                 ;
   20 65 76 69 72 0116E                    ;
20 74 6F 6E 20 6C 6C 69 77 20 6B 73 69 64 28 01173    .ASCII  \(disk will not be corrupted).!/!14* 2. P\  ;
21 2E 29 64 65 74 70 75 72 72 6F 63 20 65 62 01182                 ;
   50 20 2E 32 20 2A 34 31 21 2F 01191                             ;
4F 54 53 20 4E 55 52 22 20 6E 69 20 68 73 75 0119B    .ASCII  \ush in 'RUN STOP  or  LOAD  switch.!/!13\  ;
77 73 20 22 44 41 4F 4C 22 20 72 6F 20 22 50 011AA                 ;
   33 31 21 2F 21 2E 68 63 74 69 011B9                             ;
22 20 74 75 6F 20 68 73 75 50 20 2E 33 20 2A 011C3    .ASCII  \* 3. Push out  WRITE PROT  switch (not !\  ;
69 77 73 20 22 54 4F 52 50 20 45 54 49 52 57 011D2                 ;
   6C 20 74 6F 6E 28 20 68 63 74 011E1                             ;
20 2A 33 31 21 2F 21 2E 29 64 65 74 68 67 69 011EB    .ASCII  \ighted).!/!13* 4. Wait for  READY  indic\  ;
45 52 22 20 72 6F 66 20 74 69 61 57 20 2E 34 011FA                 ;
   63 69 64 6E 69 20 22 59 44 41 01209                             ;
   2E 74 68 67 69 6C 20 6F 74 20 72 6F 74 61 01213    .ASCII  \ator to light.\          ;
70 6F 6C 66 20 74 72 65 73 6E 49 20 2E 31 6A 01221 P.AEQ:  .ASCII  \j1. Insert floppy diskette in drive and \  ;
20 6E 69 20 65 74 74 65 6B 73 69 64 20 79 70 01230                 ;
   20 64 6E 61 20 65 76 69 72 64 0123F                             ;
31 21 2F 21 2E 72 6F 6F 64 20 65 73 6F 6C 63 01249    .ASCII  \close door.!/!13*    Current contents of\  ;
63 20 74 6E 65 72 72 75 43 20 20 20 20 2A 33 01258                 ;
   66 6F 20 73 74 6E 65 74 6E 6F 01267                             ;
42 20 59 41 4D 20 65 74 74 65 6B 73 69 64 20 01271    .ASCII  \ diskette MAY BE DESTROYED.\          ;
   2E 44 45 59 4F 52 54 53 45 44 20 45 01280                       ;
20 65 70 61 74 20 74 6E 75 6F 4D 20 2E 31 C7 0128C P.AER:  .ASCII  <199>\1. Mount tape on drive. Tape must \   ;
20 65 70 61 54 20 2E 65 76 69 72 64 20 6E 6F 0129B                 ;
   20 74 73 75 6D 012AA                    ;
67 6E 69 72 2D 65 74 69 72 77 20 65 76 61 68 012AF    .ASCII  \have write-ring.!/!13*    Current conten\  ;
72 72 75 43 20 20 20 20 2A 33 31 21 2F 21 2E 012BE                 ;
   6E 65 74 6E 6F 63 20 74 6E 65 012CD                             ;
20 59 41 4D 20 65 70 61 74 20 66 6F 20 73 74 012D7    .ASCII  \ts of tape MAY BE DESTROYED.!/!13* 2. Pu\  ;
2F 21 2E 44 45 59 4F 52 54 53 45 44 20 45 42 012E6                 ;
   75 50 20 2E 32 20 2A 33 31 21 012F5                             ;
20 72 6F 20 22 44 4F 4C 22 20 6E 69 20 68 73 012FF    .ASCII  \sh in 'LOD' or 'LOAD  switch to load tap\  ;
74 20 68 63 74 69 77 73 20 22 44 41 4F 4C 22 0130E                 ;
   70 61 74 20 64 61 6F 6C 20 6F 0131D                             ;
73 75 50 20 2E 33 20 2A 33 31 21 2F 21 2E 65 01327    .ASCII  \e.!/!13* 3. Push in  ONL  or  ONLINE  sw\  ;
22 20 72 6F 20 22 4C 4E 4F 22 20 6E 69 20 68 01336                 ;
   77 73 20 22 45 4E 49 4C 4E 4F 01345                             ;
   2E 68 63 74 69 0134F    .ASCII  \itch.\                 ;
55 52 22 20 6E 69 20 68 73 75 50 20 2E 31 80 01354 P.AES:  .ASCII  <128>\1. Push in  RUN STOP  switch.!/!13\   ;
2E 68 63 74 69 77 73 20 22 50 4F 54 53 20 4E 01363                 ;
   33 31 21 2F 21 01372                    ;
22 20 74 75 6F 20 68 73 75 50 20 2E 32 20 2A 01377    .ASCII  \* 2. Push out  WRITE PROT  switch (not !\  ;
69 77 73 20 22 54 4F 52 50 20 45 54 49 52 57 01386                 ;
   6C 20 74 6F 6E 28 20 68 63 74 01395                             ;
20 2A 33 31 21 2F 21 2E 29 64 65 74 68 67 69 0139F    .ASCII  \ighted).!/!13* 3. Wait for  READY  indic\  ;
```

```
45  52  22  20  72  6F  66  20  74  69  61  57  20  2E  33  013AE             ;
      63  69  64  6E  69  20  22  59  44  41  013BD                           ;
    2E  74  68  67  69  6C  20  6F  74  20  72  6F  74  61  013C7    .ASCII  \ator to light.\          ;
4F  4C  22  20  6E  69  20  68  73  75  50  20  2E  31  7C  013D5 P.AET:  .ASCII  \|1. Push in  LOAD  switch.!/!13* 2. Push\  ;
31  21  2F  21  2E  68  63  74  69  77  73  20  22  44  41  013E4             ;
      68  73  75  50  20  2E  32  20  2A  33  013F3                           ;
4F  52  50  20  45  54  49  52  57  22  20  74  75  6F  20  013FD    .ASCII  \ out  WRITE PROT' switch (not lighted).!\  ;
20  74  6F  6E  28  20  68  63  74  69  77  73  20  22  54  0140C             ;
      21  2E  29  64  65  74  68  67  69  6C  0141B                           ;
66  20  74  69  61  57  20  2E  33  20  2A  33  31  21  2F  01425    .ASCII  \/!13* 3. Wait for  READY  indicator to l\  ;
69  64  6E  69  20  22  59  44  41  45  52  22  20  72  6F  01434             ;
      6C  20  6F  74  20  72  6F  74  61  63  01443                           ;
     2E  74  68  67  69  0144D    .ASCII  \ight.\                 ;
6F  6D  65  72  20  74  72  65  73  6E  49  20  2E  31  BF  01452 P.AEU:  .ASCII  <191>\1. Insert removable disk in drive!\   ;
64  20  6E  69  20  6B  73  69  64  20  65  6C  62  61  76  01461             ;
      21  65  76  69  72  01470                  ;
69  20  68  73  75  50  20  2E  32  20  2A  33  31  21  2F  01475    .ASCII  \/!13* 2. Push in  RUN  switch!/!13* 3. P\  ;
21  68  63  74  69  77  73  20  22  4E  55  52  22  20  6E  01484             ;
      50  20  2E  33  20  2A  33  31  21  2F  01493                           ;
50  20  65  74  69  72  57  20  74  75  6F  20  68  73  75  0149D    .ASCII  \ush out Write Protect  REMOVE  and  FIXE\  ;
22  45  56  4F  4D  45  52  22  20  74  63  65  74  6F  72  014AC             ;
      45  58  49  46  22  20  64  6E  61  20  014BB                           ;
69  77  73  20  20  20  20  2A  33  31  21  2F  21  22  44  014C5    .ASCII  \D'!/!13*    switch (unlighted)!/!13* 4. \  ;
29  64  65  74  68  67  69  6C  6E  75  28  20  68  63  74  014D4             ;
      20  2E  34  20  2A  33  31  21  2F  21  014E3                           ;
20  22  4E  55  52  22  20  72  6F  66  20  74  69  61  57  014ED    .ASCII  \Wait for  RUN  light to stop blinking\     ;
62  20  70  6F  74  73  20  6F  74  20  74  68  67  69  6C  014FC             ;
      67  6E  69  6B  6E  69  6C  0150B                ;
72  64  20  70  75  20  72  65  77  6F  50  20  2E  31  36  01512 P.AEV:  .ASCII  \61. Power up drive by pressing (1/0) swi\  ;
67  6E  69  73  73  65  72  70  20  79  62  20  65  76  69  01521             ;
      69  77  73  20  29  30  2F  31  28  20  01530                           ;
2F  21  29  31  28  20  4E  4F  20  6F  74  20  68  63  74  0153A    .ASCII  \tch to ON (1)!/\          ;
```

```
        $MODULE=        P.AAA
        CRD$GT_SPACE_AND_SPACE==
                P.AAB
        CRD$GT_NEW_LINE==    P.AAC
        CRD$GT_TAB==        P.AAD
        CRD$GT_SLASH==       P.AAE
        CRD$GT_UNKNOWN==     P.AAF
        CRD$GT_2_TABS==      P.AAG
        CRD$GT_MENU==        P.AAH
        CRD$GT_AUTO==        P.AAI
        CRD$GT_TESTING_DEVICE==
                P.AAJ
        CRD$GT_TESTING_STARTED==
                P.AAK
        CRD$GT_RUNTIME==     P.AAL
        CRD$GT_FAIL==        P.AAM
        CRD$GT_PASS==        P.AAN
        CRD$GT_CRD_NO_ERRORS==
                P.AAO
        CRD$GT_CRD_ERRORS_COMPLETE==
                P.AAP
        CRD$GT_CRD_BAD_DEVICE==
```

```
            P.AAQ
        CRD$GT_CRD_INCOMPLETE==
            P.AAR
        CRD$GT_CRD_DEV_UNTESTED==
            P.AAS
        CRD$GT_CRD_EVSBA_ERROR==
            P.AAT
        CRD$GT_CRD_EVSBB_ERROR==
            P.AAU
        CRD$GT_CRD_FATAL_ERROR==
            P.AAV
        CRD$GT_INTERNAL_ERROR_ABORT==
            P.AAW
        CRD$GT_CONTROL_C_ABORT==
            P.AAX
        CRD$GT_CRD_END_MESSAGE==
            P.AAY
        CRD$GT_EXIT_TO_CONSOLE==
            P.AAZ
        CRD$GT_EXIT_TO_CLI==P.ABA
        CRD$GT_PRESS_RETURN==
            P.ABB
        CRD$GT_PRESS_RETURN_MM==
            P.ABC
        CRD$GT_INV_BASE-SYSTEM==
            P.ABD
        CRD$GT_AUTOSIZER_ERROR==
            P.ABE
        CRD$GT_DEV_UNAVAILABLE==
            P.ABF
        CRD$GT_DEV_UNAVAILABLE_2==
            P.ABG
        CRD$GT_CANNOT_LOAD_DIAG==
            P.ABH
        CRD$GT_START_ERROR==P.ABI
        CRD$GT_PREP_ERROR_SPACES==
            P.ABJ
        CRD$GT_INTERNAL_ERROR==
            P.ABK
        CRD$GT_DS_COMMAND_OUT==
            P.ABL
        CRD$GT_SAME_LINE_MESSAGE==
            P.ABM
        CRD$GT_NEW LINE_MESSAGE==
            P.ABN
        CRD$GT_STAR_LINE_PREFIX_MESSAGE-=
            P.ABO
        CRD$GT_STAR_LINE_SUFFIX_MESSAGE==
            P.ABP
        CRD$GT_SUMMARY_TITLE==
            P.ABQ
        CRD$GT_ALL_FAILURES=-
            P.ABR
        CRD$GT ALL_PASSES== P.ABS
        CRD$GT_ALL_INC_HDWR_PREP==
```

```
                P.ABT
        CRD$GT_ALL_OTHERS== P.ABU
        CRD$GT_INIT_ALL_PASSES==
                P.ABV
        MEN$GT_MENUTEST_BEGIN==
                P.ABW
        MEN$GT_AUTOSIZER_BEGIN==
                P.ABX
        MEN$GT_AUTOSIZER_END==
                P.ABY
        MEN$GT_FTMTITLE==   P.ABZ
        MEN$GT_FTMENU_SELECTION==
                P.ACA
        MEN$GT_FTMSEL_EXIT==P.ACB
        MEN$GT_FTMSEL_PREP==P.ACC
        MEN$GT_FTMSEL_TEST==P.ACD
        MEN$GT_FTMSEL_SYSTEM_HDWR==
                P.ACE
        MEN$GT_FTMSEL_FNCTST_ALL==
                P.ACF
        MEN$GT_FTM_CHOICE== P.ACG
        MEN$GT_FTMPROMPT==  P.ACH
        MEN$GT_TMTITLE==    P.ACI
        MEN$GT_TMENU_SELECTION==
                P.ACJ
        MEN$GT_TMENU_TSTALL==
                P.ACK
        MEN$GT_TM_CHOICE==  P.ACL
        MEN$GT_TMPROMPT==   P.ACM
        MEN$GT_TM_MSG==     P.ACN
        MEN$GT_PRESS_RETURN_TM==
                P.ACO
        MEN$GT_CNTLC_MENU_TITLE==
                P.ACP
        MEN$GT_CNTLC_MENU_SELECTION==
                P.ACQ
        MEN$GT_CNTLC_SEL_EXIT==
                P.ACB
        MEN$GT_CNTLC_SEL_CONTINUE==
                P.ACR
        MEN$GT_CNTLC_SEL_FTMENU==
                P.ACS
        MEN$GT_CNTLCM_CHOICE==
                P.ACT
        MEN$GT_CNTLC_MENU_PROMPT==
                P.ACU
        MEN$GT_CNTLC_CONTINUING==
                P.ACV
        MEN$GT_TSTCLA_CPU== P.ACW
        MEN$GT_TSTCLA_MEMORY==
                P.ACX
        MEN$GT_TSTCLA_DISK==P.ACY
        MEN$GT_TSTCLA_TAPE==P.ACZ
        MEN$GT_TSTCLA_COMM==P.ADA
        MEN$GT_TSTCLA_REAL==P.ADB
```

```
        MEN$GT_TSTCLA_TERM==P.ADC
        MEN$GT_TSTCLA_PRINTER==
              P.ADD
        MEN$GT_TSTCLA_CARD==P.ADE
        MEN$GT_TSTCLA_IOADAP==
              P.ADF
        MEN$GT_TSTCLA_NULLNAME==
              P.ADG
        MEN$GT_TSTCLA_ALL== P.ADG
        MEN$GT_SYSHDWR_TITLE==
              P.ADH
        MEN$GT_SYSTEM_HDWR_TEXT==
              P.ADI
        MEN$GT_SUPPORTED==  P.ADJ
        MEN$GT_NOSUPPORT==  P.ADK
        MEN$GT_END_OF_LIST==P.ADL
        MEN$GT_CTRLC_FNCTST==
              P.ADM
        MEN$GT_FUNCTEST_BEGIN==
              P.ADN
        MEN$GT_RUNTIME_TOTAL==
              P.ADO
        MEN$GT_FNCTST_CMPL_NOERR==
              P.ADP
        MEN$GT_FNCTST_INCMPL_ERR==
              P.ADQ
        MEN$GT_FNCTST_CMPL_ERR==
              P.ADR
        MEN$GT_FNCTST_INCMPL_NOERR==
              P.ADS
        MEN$GT_ASSISTANCE_MSG==
              P.ADT
        MEN$GT_MENUTEST_ABORTED==
              P.ADU
        MEN$GT_OPERATOR_ABORT==
              P.ADV
        MEN$GT_OPERATOR_STOP==
              P.ADW
        MEN$GT_FNCTST_TESTSTRT_TEXT==
              P.ADX
        MEN$GT_NULL==       P.ADY
        MEN$GT_CALL_FLDSRV==P.ADZ
        MEN$GT_INVOPERINPUT==
              P.AEA
        MEN$GT_FAILED_HDWR_NAME==
              P.AEB
        MEN$GT_PREP_ERROR_TEXT==
              P.AEC
        MEN$GT_PREP_ERROR_TEXT NO USER-=
              P.AED
        MEN$GT_FAILED_ONL_AUTOSIZER-=
              P.AEE
        MEN$GT_SCRMEDIA MSG=-
              P.AEF
        MEN$GT_SCRMEDIA HDWR -
```

```
                P.AEG
        MEN$GT_SCRMEDIA_HDWR_END==
                P.AEH
        MEN$GT_SCRMEDIA_PROMPT==
                P.AEI
        MEN$GT_NOALL_NONPGMEM==
                P.AEJ
        MEN$GT_SUPPORT_FILE_NOT_FOUND==
                P.AEK
        MEN$GT_SUPPORT_FILE_LOAD_ERR==
                P.AEL
        MEN$GT_DEVTSTPREP_NAME==
                P.AEM
        MEN$GT_DEVTSTPREP_TEXT==
                P.AEN
        MEN$GT_DEVTSTPREP_NULL==
                P.AEO
        MEN$GT_DEVTSTPREP_1==
                P.AEP
        MEN$GT_DEVTSTPREP_2==
                P.AEQ
        MEN$GT_DEVTSTPREP_3==
                P.AER
        MEN$GT_DEVTSTPREP_4==
                P.AES
        MEN$GT_DEVTSTPREP_5==
                P.AET
        MEN$GT_DEVTSTPREP_6==
                P.AEU
        MEN$GT_DEVTSTPREP_7==
                P.AEV


;               PSECT SUMMARY
;
; Name     Bytes          Attributes
;
;  DATA          5449  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK           132  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ --- -------- ----- ----------------
$ASCIC    Macro    Lib02   164     165     166     167     168     169     178     179     185     186
              187     188     189     196     197     198     200     202     204     205
              206     207     208     214     215     216     222     223     228     229
              236     237     238     239     241     243     250     256     257     258
              259     265     266     267     269     270     271     283     292     295
              302     305     308     311     314     317     320     324     327     334
              337     340     343     346     349     352     358     361     366     369
              372     375     378     435     440     443     446     449     457     463
              466     469     472     475     478     481     484     487     490     493
              496     501     504     507     510     514     517     524     532     535
              538     545     548     551     559     562     565     568     574     578
              584     589     594     601
$CRD_LITERALS   Macro    Lib03   126
$EQULST    Macro    Lib04   126     127
$ER    Comptime   147
$MEN_LITERALS   Macro    Lib03   127
$MODULE    Bind        147
AUTO$K_EXIT_AUTOSIZER_ERROR Literal    126
AUTO$K_EXIT_CONTROL_C  Literal    126
AUTO$K_EXIT_DUMMY_2  Literal    126
AUTO$K_EXIT_DUMMY_3  Literal    126
AUTO$K_EXIT_ERRORS_COMPLETE Literal    126
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal    126
AUTO$K_EXIT_INTERNAL_ERROR Literal    126
AUTO$K_EXIT_INV_BASE_SYSTEM Literal    126
AUTO$K_EXIT_LAST_REASON  Literal    126   126
AUTO$K_EXIT_NOERRORS_COMPLETE Literal    126
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal    126
AUTO$K_EXIT_NOERRORS_PREP Literal    126
CRD$GA_CRD_MESSAGE_BUFFER Global     155 Lib03e
CRD$GT_2_TABS   External Lib03
    GlobBind   169
CRD$GT_ALL_FAILURES  External Lib03
    GlobBind   266
CRD$GT_ALL_INC_HDWR_PREP External Lib03
    GlobBind   268
CRD$GT_ALL_OTHERS  External Lib03
    GlobBind   270
CRD$GT_ALL_PASSES  External Lib03
    GlobBind   267
CRD$GT_AUTO   External Lib03
    GlobBind   179
CRD$GT_AUTOSIZER_ERROR  External Lib03
    GlobBind   229
CRD$GT_CANNOT_LOAD_DIAG  External Lib03
    GlobBind   238
CRD$GT_CONTROL_C_ABORT  External Lib03
    GlobBind   208
CRD$GT_CRD_BAD_DEVICE  External Lib03
    GlobBind   198
CRD$GT_CRD_DEV_UNTESTED  External Lib03
    GlobBind   202
CRD$GT_CRD_END_MESSAGE  External Lib03
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    GlobBind   214
CRD$GT_CRD_ERRORS_COMPLETE External Lib03
    GlobBind   197
CRD$GT_CRD_EVSBA_ERROR  External Lib03
    GlobBind   204
CRD$GT_CRD_EVSBB_ERROR  External Lib03
    GlobBind   205
CRD$GT_CRD_FATAL_ERROR  External Lib03
    GlobBind   206
CRD$GT_CRD_INCOMPLETE  External Lib03
    GlobBind   199
CRD$GT_CRD_NO_ERRORS  External Lib03
    GlobBind   196
CRD$GT_DEV_UNAVAILABLE  External Lib03
    GlobBind   236
CRD$GT_DEV_UNAVAILABLE_2 External Lib03
    GlobBind   237
CRD$GT_DS_COMMAND_OUT  External Lib03
    GlobBind   250
CRD$GT_EXIT_TO_CLI  External Lib03
    GlobBind   216
CRD$GT_EXIT_TO_CONSOLE  External Lib03
    GlobBind   215
CRD$GT_FAIL   External Lib03
    GlobBind   188
CRD$GT_INIT_ALL_PASSES  External Lib03
    GlobBind   271
CRD$GT_INTERNAL_ERROR  External Lib03
    GlobBind   243
CRD$GT_INTERNAL_ERROR_ABORT External Lib03
    GlobBind   207
CRD$GT_INV_BASE_SYSTEM  External Lib03
    GlobBind   228
CRD$GT_MENU   External Lib03
    GlobBind   178
CRD$GT_NEW_LINE   External Lib03
    GlobBind   165
CRD$GT_NEW_LINE_MESSAGE  External Lib03
    GlobBind   257
CRD$GT_PASS   External Lib03
    GlobBind   189
CRD$GT_PREP_ERROR_SPACES External Lib03
    GlobBind   241
CRD$GT_PRESS_RETURN  External Lib03
    GlobBind   222
CRD$GT_PRESS_RETURN_MM  External Lib03
    GlobBind   223
CRD$GT_RUNTIME   External Lib03
    GlobBind   187
CRD$GT_SAME_LINE_MESSAGE External Lib03
    GlobBind   256
CRD$GT_SLASH   External Lib03
    GlobBind   167
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------- -------- ----- ---------------
CRD$GT_SPACE_AND_SPACE  External Lib03
    GlobBind   164
CRD$GT_START_ERROR  External Lib03
    GlobBind   239
CRD$GT_STAR_LINE_PREFIX_MESSAGE External Lib03
    GlobBind   258
CRD$GT_STAR_LINE_SUFFIX_MESSAGE External Lib03
    GlobBind   259
CRD$GT_SUMMARY_TITLE  External Lib03
    GlobBind   264
CRD$GT_TAB   External Lib03
    GlobBind   166
CRD$GT_TESTING_DEVICE  External Lib03
    GlobBind   185
CRD$GT_TESTING_STARTED  External Lib03
    GlobBind   186
CRD$GT_UNKNOWN   External Lib03
    GlobBind   168
CRD$K_ACTION_EQL_DO_NOTHING Literal    126
CRD$K_ACTION_RANGE_ERROR Literal    126
CRD$K_ADVANCE_DIAGNOSTIC Literal    126
CRD$K_ADVANCE_GENERAL_COM Literal    126
CRD$K_ADVANCE_STATE  Literal    126
CRD$K_ALLOCATION_ERROR  Literal    126
CRD$K_ASSIGN_PTABLE_PTRS Literal    126
CRD$K_AUTOSIZER_ERROR  Literal    126
CRD$K_AUTOSIZER_SET_FLAGS Literal    126
CRD$K_BAD_ACTION_NUMBER  Literal    126
CRD$K_BAD_TYPECODE_ERROR Literal    126
CRD$K_BASE_SYSTEM_IDC  Literal    126
CRD$K_BASE_SYSTEM_LESI  Literal    126
CRD$K_BASE_SYSTEM_NULL  Literal    126
CRD$K_BASE_SYSTEM_UDA_0  Literal    126
CRD$K_BASE_SYSTEM_UDA_1  Literal    126
CRD$K_BASE_SYSTEM_UDA_2  Literal    126
CRD$K_BASE_SYSTEM_UDA_3  Literal    126
CRD$K_CRD_INITIALIZATION Literal    126
CRD$K_CRD_MESSAGE_BUFFER_LEN Literal  Lib03   155
CRD$K_CREATE_HST  Literal    126
CRD$K_DATA_LENGTH_ERROR  Literal    126
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal    126
CRD$K_DDB_BLINK   Literal    126
CRD$K_DDB_FLINK   Literal    126
CRD$K_DDB_LAST_ENTRY  Literal    126
CRD$K_DDB_MEMORY_SIZE  Literal    126
CRD$K_DDB_MENU_DEVICE_NUMB Literal    126
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal    126
CRD$K_DDB_MENU_SUPPORT_SIZE Literal    126
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal    126
CRD$K_DDB_PTABLE_ENTRY  Literal    126
CRD$K_DDB_STATUS  Literal    126
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal    126
CRD$K_DEVICE_NAME  Literal    126
```

Symbol	Type	Defined	Referenced ...
CRD$K_DIAGNOSTIC_ERROR	Literal	126	
CRD$K_DIAGNOSTIC_NAME	Literal	126	
CRD$K_DONE_GENERAL_COMS	Literal	126	
CRD$K_DO_NOTHING	Literal	126	
CRD$K_DUMMY_10	Literal	126	
CRD$K_DUMMY_11	Literal	126	
CRD$K_DUMMY_12	Literal	126	
CRD$K_DUMMY_13	Literal	126	
CRD$K_DUMMY_3	Literal	126	
CRD$K_DUMMY_4	Literal	126	
CRD$K_DUMMY_5	Literal	126	
CRD$K_DUMMY_6	Literal	126	
CRD$K_DUMMY_7	Literal	126	
CRD$K_DUMMY_8	Literal	126	
CRD$K_DUMMY_9	Literal	126	
CRD$K_EXIT_CRD	Literal	126	
CRD$K_HOOK_POINT	Literal	126	
CRD$K_HS_INTERNAL_ERROR	Literal	126	
CRD$K_IDC_EVDLB	Literal	126	
CRD$K_IDC_EVDLC	Literal	126	
CRD$K_IDC_EVDLD	Literal	126	
CRD$K_IDC_EVRAD_0	Literal	126	
CRD$K_IDC_EVRAD_1	Literal	126	
CRD$K_IDC_LAST_DIAGNOSTIC	Literal	126	
CRD$K_INTERNAL_ERROR	Literal	126	
CRD$K_LAST_ACTION_ROUTINE	Literal	126	
CRD$K_LAST_ENTRY	Literal	126	
CRD$K_LAST_FUNCTION	Literal	126	
CRD$K_LAST_HOOK_POINT	Literal	126	
CRD$K_LAST_INTERNAL_ERROR	Literal	126	
CRD$K_LAST_TYPE	Literal	126	
CRD$K_LESI_ENWCA	Literal	126	
CRD$K_LESI_EVRMB_0	Literal	126	
CRD$K_LESI_EVRMB_1	Literal	126	
CRD$K_LESI_LAST_DIAGNOSTIC	Literal	126	
CRD$K_LEVEL_4_PASSED	Literal	126	
CRD$K_LOAD_AUTOSIZER	Literal	126	
CRD$K_LOAD_ERROR	Literal	126	
CRD$K_LOAD_GENERAL_COM	Literal	126	
CRD$K_LOAD_LOAD_COM	Literal	126	
CRD$K_LOAD_SELECT_COM	Literal	126	
CRD$K_LOAD_SET_EVENT_COM	Literal	126	
CRD$K_LOAD_SET_FLAGS_COM	Literal	126	
CRD$K_LOAD_START_COM	Literal	126	
CRD$K_LOAD_SUP_FILE	Literal	126	
CRD$K_MM_INTERNAL_ERROR	Literal	126	
CRD$K_NEW_LINE	Literal	126	
CRD$K_PREPARATION_ERROR	Literal	126	
CRD$K_PREPARATION_ERROR_TEXT	Literal	126	
CRD$K_RUNTIME	Literal	126	
CRD$K_SAME_LINE	Literal	126	
CRD$K_SET_EVENT_ARGS	Literal	126	
CRD$K_SET_FLAGS_ARGS	Literal	126	

Symbol	Type	Defined	Referenced ...
CRD$K_SET_UP_DIAGNOSTIC	Literal	126	
CRD$K_START	Literal	126	
CRD$K_START_AUTOSIZER	Literal	126	
CRD$K_START_ERROR	Literal	126	
CRD$K_START_QUALIFIERS	Literal	126	
CRD$K_STAR_LINE	Literal	126	
CRD$K_STATE_ADVANCE_DIAGNOSTIC	Literal	126	
CRD$K_STATE_ADVANCE_GENERAL_COM	Literal	126	
CRD$K_STATE_ASSIGN_PTABLE_PTRS	Literal	126	
CRD$K_STATE_AUTOSIZER_ERROR	Literal	126	
CRD$K_STATE_AUTOSIZER_RUNNING	Literal	126	
CRD$K_STATE_AUTOSIZER_SET_FLAGS	Literal	126	
CRD$K_STATE_DIAGNOSTIC_ERROR	Literal	126	
CRD$K_STATE_DIAGNOSTIC_RUNNING	Literal	126	
CRD$K_STATE_DONE_DIAGNOSTICS	Literal	126	
CRD$K_STATE_DONE_GENERAL_COMS	Literal	126	
CRD$K_STATE_DUMMY_1	Literal	126	
CRD$K_STATE_DUMMY_10	Literal	126	
CRD$K_STATE_DUMMY_12	Literal	126	
CRD$K_STATE_DUMMY_13	Literal	126	
CRD$K_STATE_DUMMY_2	Literal	126	
CRD$K_STATE_DUMMY_3	Literal	126	
CRD$K_STATE_DUMMY_4	Literal	126	
CRD$K_STATE_DUMMY_6	Literal	126	
CRD$K_STATE_DUMMY_7	Literal	126	
CRD$K_STATE_DUMMY_8	Literal	126	
CRD$K_STATE_EXIT_CRD	Literal	126	
CRD$K_STATE_INTERNAL_ERROR	Literal	126	
CRD$K_STATE_LAST_STATE	Literal	126	
CRD$K_STATE_LEVEL_4_PASSED	Literal	126	
CRD$K_STATE_LOAD_AUTOSIZER	Literal	126	
CRD$K_STATE_LOAD_ERROR	Literal	126	
CRD$K_STATE_LOAD_GENERAL_COM	Literal	126	
CRD$K_STATE_LOAD_LOAD_COM	Literal	126	
CRD$K_STATE_LOAD_SELECT_COM	Literal	126	
CRD$K_STATE_LOAD_SET_EVENT_COM	Literal	126	
CRD$K_STATE_LOAD_SET_FLAGS_COM	Literal	126	
CRD$K_STATE_LOAD_START_COM	Literal	126	
CRD$K_STATE_PREPARATION_ERROR	Literal	126	
CRD$K_STATE_SET_UP_DIAGNOSTIC	Literal	126	
CRD$K_STATE_START	Literal	126	
CRD$K_STATE_START_AUTOSIZER	Literal	126	
CRD$K_STATE_START_ERROR	Literal	126	
CRD$K_TEST_START_TEXT	Literal	126	
CRD$K_TQ_INTERNAL_ERROR	Literal	126	
CRD$K_TYPECODE	Literal	126	
CRD$K_UDA_0_EVDLB	Literal	126	
CRD$K_UDA_0_EVDLC	Literal	126	
CRD$K_UDA_0_EVDLD	Literal	126	
CRD$K_UDA_0_EVRLA_0	Literal	126	
CRD$K_UDA_0_LAST_DIAGNOSTIC	Literal	126	
CRD$K_UDA_1_EVDLB	Literal	126	
CRD$K_UDA_1_EVDLC	Literal	126	

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_UDA_1_EVDLD  Literal    126
CRD$K_UDA_1_EVRLA_0  Literal    126
CRD$K_UDA_1_EVRLA_1  Literal    126
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal    126
CRD$K_UDA_2_EVDLB  Literal    126
CRD$K_UDA_2_EVDLC  Literal    126
CRD$K_UDA_2_EVDLD  Literal    126
CRD$K_UDA_2_EVRLA_0  Literal    126
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal    126
CRD$K_UDA_3_EVDLB  Literal    126
CRD$K_UDA_3_EVDLC  Literal    126
CRD$K_UDA_3_EVDLD  Literal    126
CRD$K_UDA_3_EVRLA_0  Literal    126
CRD$K_UDA_3_EVRLA_1  Literal    126
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal    126
GET1ST_    Macro    Lib04   126     127
GET2ND_    Unbound          126     127
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal    126
HS$K_ACTION_ADVANCE_GENERAL_COM Literal    126
HS$K_ACTION_ADVANCE_STATE Literal    126
HS$K_ACTION_CHANGE_TABLE Literal    126
HS$K_ACTION_DIAGNOSTIC_ERROR Literal    126
HS$K_ACTION_DONE_GENERAL_COMS Literal    126
HS$K_ACTION_DO_NOTHING  Literal    126
HS$K_ACTION_DUMMY_3  Literal    126
HS$K_ACTION_DUMMY_4  Literal    126
HS$K_ACTION_DUMMY_5  Literal    126
HS$K_ACTION_DUMMY_6  Literal    126
HS$K_ACTION_INIT_HS  Literal    126
HS$K_ACTION_INTERNAL_ERROR Literal    126
HS$K_ACTION_LAST_ACTION  Literal    126
HS$K_ACTION_LOAD_ERROR  Literal    126
HS$K_ACTION_LOAD_GENERAL_COM Literal    126
HS$K_ACTION_LOAD_LOAD_COM Literal    126
HS$K_ACTION_LOAD_SELECT_COM Literal    126
HS$K_ACTION_LOAD_SET_EVENT_COM Literal    126
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal    126
HS$K_ACTION_LOAD_START_COM Literal    126
HS$K_ACTION_PREPARATION_ERROR Literal    126
HS$K_ACTION_START_ERROR  Literal    126
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal    126
HS$K_STATE_ADVANCE_GENERAL_COM Literal    126
HS$K_STATE_CHANGE_TABLE  Literal    126
HS$K_STATE_DIAGNOSTIC_ERROR Literal    126
HS$K_STATE_DIAGNOSTIC_RUNNING Literal    126
HS$K_STATE_DONE_GENERAL_COMS Literal    126
HS$K_STATE_DUMMY_1  Literal    126
HS$K_STATE_DUMMY_2  Literal    126
HS$K_STATE_DUMMY_3  Literal    126
HS$K_STATE_DUMMY_4  Literal    126
HS$K_STATE_DUMMY_6  Literal    126
HS$K_STATE_INIT_HS  Literal    126
HS$K_STATE_INTERNAL_ERROR Literal    126
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_STATE_LAST_STATE  Literal    126
HS$K_STATE_LOAD_ERROR  Literal    126
HS$K_STATE_LOAD_GENERAL_COM Literal    126
HS$K_STATE_LOAD_LOAD_COM Literal    126
HS$K_STATE_LOAD_SELECT_COM Literal    126
HS$K_STATE_LOAD_SET_EVENT_COM Literal    126
HS$K_STATE_LOAD_SET_FLAGS_COM Literal    126
HS$K_STATE_LOAD_START_COM Literal    126
HS$K_STATE_PREPARATION_ERROR Literal    126
HS$K_STATE_START_ERROR  Literal    126
MEN$GK_TEST_QUEUE  Literal    127
MEN$GT_ASSISTANCE_MSG  External Lib03
    GlobBind   480
MEN$GT_AUTOSIZER_BEGIN  External Lib03
    GlobBind   291
MEN$GT_AUTOSIZER_END  External Lib03
    GlobBind   294
MEN$GT_CALL_FLDSRV  External Lib03
    GlobBind   500
MEN$GT_CNTLCM_CHOICE  External Lib03
    GlobBind   371
MEN$GT_CNTLC_CONTINUING  External Lib03
    GlobBind   377
MEN$GT_CNTLC_MENU_PROMPT External Lib03
    GlobBind   374
MEN$GT_CNTLC_MENU_SELECTION External Lib03
    GlobBind   360
MEN$GT_CNTLC_MENU_TITLE  External Lib03
    GlobBind   357
MEN$GT_CNTLC_SEL_CONTINUE External Lib03
    GlobBind   365
MEN$GT_CNTLC_SEL_EXIT  External Lib03
    GlobBind   363
MEN$GT_CNTLC_SEL_FTMENU  External Lib03
    GlobBind   368
MEN$GT_CTRLC_FNCTST  External Lib03
    GlobBind   456
MEN$GT_DEVTSTPREP_1  External Lib03
    GlobBind   567
MEN$GT_DEVTSTPREP_2  External Lib03
    GlobBind   573
MEN$GT_DEVTSTPREP_3  External Lib03
    GlobBind   577
MEN$GT_DEVTSTPREP_4  External Lib03
    GlobBind   583
MEN$GT_DEVTSTPREP_5  External Lib03
    GlobBind   588
MEN$GT DEVTSTPREP 6  External Lib03
    GlobBind   593
MEN$GT DEVTSTPREP 7  External Lib03
    GlobBind   600
MEN$GT_DEVTSTPREP_NAME  External Lib03
    GlobBind   558
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$GT_DEVTSTPREP_NULL  External Lib03
    GlobBind   564
MEN$GT_DEVTSTPREP_TEXT  External Lib03
    GlobBind   561
MEN$GT_END_OF_LIST  External Lib03
    GlobBind   448
MEN$GT_FAILED_HDWR_NAME  External Lib03
    GlobBind   506
MEN$GT_FAILED_ONL_AUTOSIZER External Lib03
    GlobBind   516
MEN$GT_FNCTST_CMPL_ERR  External Lib03
    GlobBind   474
MEN$GT_FNCTST_CMPL_NOERR External Lib03
    GlobBind   468
MEN$GT_FNCTST_INCMPL_ERR External Lib03
    GlobBind   471
MEN$GT_FNCTST_INCMPL_NOERR External Lib03
    GlobBind   477
MEN$GT_FNCTST_TESTSTRT_TEXT External Lib03
    GlobBind   492
MEN$GT_FTMENU_SELECTION  External Lib03
    GlobBind   304
MEN$GT FTMPROMPT  External Lib03
    GlobBind   326
MEN$GT_FTMSEL_EXIT  External Lib03
    GlobBind   307   363a
MEN$GT_FTMSEL_FNCTST_ALL External Lib03
    GlobBind   319
MEN$GT_FTMSEL_PREP  External Lib03
    GlobBind   310
MEN$GT_FTMSEL_SYSTEM_HDWR External Lib03
    GlobBind   316
MEN$GT_FTMSEL_TEST  External Lib03
    GlobBind   313
MEN$GT_FTMTITLE   External Lib03
    GlobBind   301
MEN$GT_FTM_CHOICE  External Lib03
    GlobBind   323
MEN$GT_FUNCTEST_BEGIN  External Lib03
    GlobBind   462
MEN$GT INVOPERINPUT  External Lib03
    GlobBind   503
MEN$GT_MENUTEST_ABORTED  External Lib03
    GlobBind   483
MEN$GT_MENUTEST_BEGIN  External Lib03
    GlobBind   282
MEN$GT_NOALL_NONPGMEM  External Lib03
    GlobBind   544
MEN$GT_NOSUPPORT  External Lib03
    GlobBind   445
MEN$GT_NULL   External Lib03
    GlobBind   495
MEN$GT_OPERATOR_ABORT  External L'b03
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    GlobBind   486
MEN$GT_OPERATOR_STOP  External Lib03
    GlobBind   489
MEN$GT_PREP_ERROR_TEXT  External Lib03
    GlobBind   509
MEN$GT_PREP_ERROR_TEXT_NO_USER External Lib03
    GlobBind   513
MEN$GT_PRESS_RETURN_TM  External Lib03
    GlobBind   351
MEN$GT_RUNTIME_TOTAL  External Lib03
    GlobBind   465
MEN$GT_SCRMEDIA_HDWR  External Lib03
    GlobBind   531
MEN$GT_SCRMEDIA_HDWR_END External Lib03
    GlobBind   534
MEN$GT_SCRMEDIA_MSG  External Lib03
    GlobBind   523
MEN$GT_SCRMEDIA_PROMPT  External Lib03
    GlobBind   537
MEN$GT_SUPPORTED  External Lib03
    GlobBind   442
MEN$GT_SUPPORT_FILE_LOAD_ERR External Lib03
    GlobBind   550
MEN$GT_SUPPORT_FILE_NOT_FOUND External Lib03
    GlobBind   547
MEN$GT_SYSHDWR_TITLE  External Lib03
    GlobBind   434
MEN$GT_SYSTEM_HDWR_TEXT  External Lib03
    GlobBind   439
MEN$GT_TMENU_SELECTION  External Lib03
    GlobBind   336
MEN$GT_TMENU_TSTALL  External Lib03
    GlobBind   339
MEN$GT_TMPROMPT  External Lib03
    GlobBind   345
MEN$GT_TMTITLE  External Lib03
    GlobBind   333
MEN$GT_TM_CHOICE  External Lib03
    GlobBind   342
MEN$GT_TM_MSG  External Lib03
    GlobBind   348
MEN$GT_TSTCLA_ALL  External Lib03
    GlobBind   428
MEN$GT_TSTCLA_CARD  External Lib03
    GlobBind   416
MEN$GT_TSTCLA_COMM  External Lib03
    GlobBind   400
MEN$GT_TSTCLA_CPU  External Lib03
    GlobBind   384
MEN$GT_TSTCLA_DISK  External Lib03
    GlobBind   392
MEN$GT_TSTCLA_IOADAP  External Lib03
    GlobBind   420
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$GT_TSTCLA_MEMORY  External Lib03
    GlobBind   388
MEN$GT_TSTCLA_NULLNAME  External Lib03
    GlobBind   424   428a
MEN$GT_TSTCLA_PRINTER  External Lib03
    GlobBind   412
MEN$GT_TSTCLA_REAL  External Lib03
    GlobBind   404
MEN$GT_TSTCLA_TAPE  External Lib03
    GlobBind   396
MEN$GT_TSTCLA_TERM  External Lib03
    GlobBind   408
MEN$K_CNTLC_SEL_CONTINUE Literal    127
MEN$K_CNTLC_SEL_EXIT  Literal    127
MEN$K_CNTLC_SEL_FTMENU  Literal    127
MEN$K_CNTLC_SEL_LAST_ENTRY Literal    127
MEN$K_DEVTSTPREP_1  Literal    127
MEN$K_DEVTSTPREP_2  Literal    127
MEN$K_DEVTSTPREP_3  Literal    127
MEN$K_DEVTSTPREP_4  Literal    127
MEN$K_DEVTSTPREP_5  Literal    127
MEN$K_DEVTSTPREP_6  Literal    127
MEN$K_DEVTSTPREP_7  Literal    127
MEN$K_DEVTSTPREP_NULL  Literal    127
MEN$K_FTMRET_EXIT  Literal    127
MEN$K_FTMRET_FAILURE  Literal    127
MEN$K_FTMRET_MENU  Literal    127
MEN$K_FTMRET_TEST  Literal    127
MEN$K_FTMSEL_EXIT  Literal    127
MEN$K_FTMSEL_FNCTST_ALL  Literal    127
MEN$K_FTMSEL_LAST_ENTRY  Literal    127
MEN$K_FTMSEL_PREP  Literal    127
MEN$K_FTMSEL_SYSTEM_HDWR Literal    127
MEN$K_FTMSEL_TEST  Literal    127
MEN$K_HS_STATE_TABLE  Literal    126
MEN$K_MM_STATE_TABLE  Literal    126
MEN$K_PREP_ERROR_TEXT_LEN Literal    127
MEN$K_SELDEV_NUMB_START  Literal    127
MEN$K_TMENU_TSTALL_DEVNUMB Literal    127
MEN$K_TQ_STATE_TABLE  Literal    126
MEN$K_TSTCLA_ALL  Literal    127
MEN$K_TSTCLA_CARD  Literal    127
MEN$K_TSTCLA_COMM  Literal    127
MEN$K_TSTCLA_CPU  Literal    127
MEN$K_TSTCLA_DISK  Literal    127
MEN$K_TSTCLA_IO_ADAPTOR  Literal    127
MEN$K_TSTCLA_LAST_ENTRY  Literal    127
MEN$K_TSTCLA_MEMORY  Literal    127
MEN$K_TSTCLA_NULL  Literal    127
MEN$K_TSTCLA_PRINTER  Literal    127
MEN$K_TSTCLA_REAL  Literal    127
MEN$K_TSTCLA_TAPE  Literal    127
MEN$K_TSTCLA_TERM  Literal    127
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$K_TSTCTG_CPU  Literal    127
MEN$K_TSTCTG_IO_ADAPTOR  Literal    127
MEN$K_TSTCTG_IO_CONTROLLER Literal    127
MEN$K_TSTCTG_IO_UNIT  Literal    127
MEN$K_TSTCTG_MEMORY  Literal    127
MEN$K_TSTCTG_NULL  Literal    127
MEN$K_TSTTXT_DEVNAM_SZ  Literal    127
MEN$K_TSTTXT_TSTCLA_SZ  Literal    127   386     390     394     398     402     406     410     414     418     422
            426
MEN$K_TSTTXT_UUTNAM_SZ  Literal    127
MENU$K_EXIT_AUTOSIZER_ERROR Literal    126
MENU$K_EXIT_INTERNAL_ERROR Literal    126
MENU$K_EXIT_LAST_REASON  Literal    126
MENU$K_EXIT_OPERATOR_ABORT Literal    126
MENU$K_EXIT_OPERATOR_STOP Literal    126
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal    126
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal    126
MM$K_ACTION_ADVANCE_STATE Literal    126
MM$K_ACTION_AUTOSIZER_ERROR Literal    126
MM$K_ACTION_BUILD_DDBS  Literal    126
MM$K_ACTION_BUILD_HSTS  Literal    126
MM$K_ACTION_CHANGE_TABLE Literal    126
MM$K_ACTION_DONE_INITS  Literal    126
MM$K_ACTION_DO_NOTHING  Literal    126
MM$K_ACTION_DUMMY_3  Literal    126
MM$K_ACTION_DUMMY_4  Literal    126
MM$K_ACTION_DUMMY_5  Literal    126
MM$K_ACTION_DUMMY_6  Literal    126
MM$K_ACTION_DUMMY_7  Literal    126
MM$K_ACTION_DUMMY_8  Literal    126
MM$K_ACTION_INTERNAL_ERROR Literal    126
MM$K_ACTION_LAST_ACTION  Literal    126
MM$K_ACTION_LOAD_AUTOSIZER Literal    126
MM$K_ACTION_MENU_PROMPT  Literal    126
MM$K_ACTION_PRINT_MENU  Literal    126
MM$K_ACTION_PRINT_MENU_HEADING Literal    126
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal    126
MM$K_ACTION_START_AUTOSIZER Literal    126
MM$K_STATE_AUTOSIZER_ERROR Literal    126
MM$K_STATE_AUTOSIZER_RUNNING Literal    126
MM$K_STATE_BUILD_DDBS  Literal    126
MM$K_STATE_BUILD_HSTS  Literal    126
MM$K_STATE_CHANGE_TABLE  Literal    126
MM$K_STATE_DONE_INITS  Literal    126
MM$K_STATE_DUMMY_1  Literal    126
MM$K_STATE_DUMMY_2  Literal    126
MM$K_STATE_DUMMY_3  Literal    126
MM$K_STATE_DUMMY_5  Literal    126
MM$K_STATE_DUMMY_7  Literal    126
MM$K_STATE_DUMMY_8  Literal    126
MM$K_STATE_INTERNAL_ERROR Literal    126
MM$K_STATE_LAST_STATE  Literal    126
MM$K_STATE_LOAD_AUTOSIZER Literal    126
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_STATE_MENU_PROMPT  Literal    126
MM$K_STATE_PRINT_MENU  Literal    126
MM$K_STATE_PRINT_MENU_HEADING Literal    126
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal    126
MM$K_STATE_START  Literal    126
MM$K_STATE_START_AUTOSIZER Literal    126
MODNAM    Macro    Lib01   147
NUL2ND    Macro    Lib04   126     127
TQ$K_ACTION_ADVANCE_STATE Literal    126
TQ$K_ACTION_CHANGE_TABLE Literal    126
TQ$K_ACTION_DONE_TEST_QUEUE Literal    126
TQ$K_ACTION_DO_NOTHING  Literal    126
TQ$K_ACTION_DUMMY_3  Literal    126
TQ$K_ACTION_DUMMY_4  Literal    126
TQ$K_ACTION_DUMMY_5  Literal    126
TQ$K_ACTION_GET_DDB  Literal    126
TQ$K_ACTION_INIT_TQ  Literal    126
TQ$K_ACTION_INTERNAL_ERROR Literal    126
TQ$K_ACTION_LAST_ACTION  Literal    126
TQ$K_STATE_CHANGE_TABLE  Literal    126
TQ$K_STATE_DONE_TEST_QUEUE Literal    126
TQ$K_STATE_DUMMY_1  Literal    126
TQ$K_STATE_DUMMY_2  Literal    126
TQ$K_STATE_DUMMY_3  Literal    126
TQ$K_STATE_DUMMY_4  Literal    126
TQ$K_STATE_DUMMY_5  Literal    126
TQ$K_STATE_GET_DDB  Literal    126
TQ$K_STATE_INIT_TQ  Literal    126
TQ$K_STATE_INTERNAL_ERROR Literal    126
TQ$K_STATE_LAST_STATE  Literal    126


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDTEXT.B32;1
    2  Module CRDTEXT
  110  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
  113  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
  116  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
  119  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  604  Eludom CRDTEXT
```

```
 KEY TO REFERENCE TYPE FLAGS

      . Fetch
      = Store
      c Routine call
      a Address use
      @ Indirect use
      e External, external routine, or external literal declaration
      f Forward or forward routine declaration
      h Condition handler enabling
      m Map declaration
      u Undeclare declaration


;     Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923          1          0          94          00:00.1
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486        131         26          62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0       1000           00:04.2



;      COMMAND QUALIFIERS

; BLISS CRDTEXT.B32/LIST=CRDTEXT.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDTEXT.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  0 code + 5581 data bytes
; Run Time:     00:29.2
; Elapsed Time:     00:54.2
; Lines/CPU Min:     1239
; Lexemes/CPU-Min: 47445
; Memory Used: 276 pages
; Compilation Complete
```

```
;    0001  0 %TITLE '*** CRDTQACT Module'
;    0002  0 MODULE CRDTQACT (    ! Contains action routines for Test Queue
;    0003  0   IDENT = '01-00'
;    0004  0   ) =
;    0005  0 !++
;    0006  0 ! COPYRIGHT (c) 1980, 1981
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! This module contains those action routines needed by the Test Queue State Table.
;    0032  0 !
;    0033  0 ! Author:
;    0034  0 !
;    0035  0 ! Jack Stansbury
;    0036  0 !
;    0037  0 ! Creation Date:
;    0038  0 !
;    0039  0 ! July 30, 1982
;    0040  0 !
;    0041  0 ! Version:
;    0042  0 !
;    0043  0 ! 6.9
;    0044  0 !
;    0045  0 ! Modified By:
;    0046  0 !
;    0047  0 ! None
;    0048  0 !-
```

```
;    0049  0 %SBTTL 'Libraries'
;    0050  1 BEGIN     ! Begin the CRDTQACT module
;    0051  1
;    0052  1 LIBRARY
;    0053  1   '$DS';     ! Use Ds.L32 first
;    0054  1
;    0055  1 LIBRARY
;    0056  1   '$Diag';    ! Use Diag.L32 second
;    0057  1
;    0058  1 LIBRARY
;    0059  1   '$CRD';     ! Use CRD.L32 third
;    0060  1
;    0061  1 LIBRARY
;    0062  1   'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0063  1 %SBTTL 'Table of Contents'
;    0064  1
;    0065  1 FORWARD ROUTINE
;    0066  1  TQ$Init_TQ    : ADDRESSING_MODE (LONG_RELATIVE),
;    0067  1  TQ$Get_DDB    : ADDRESSING_MODE (LONG_RELATIVE),
;    0068  1  TQ$Change_Table    : ADDRESSING_MODE (LONG_RELATIVE),
;    0069  1  TQ$Done_Test_Queue  : ADDRESSING_MODE (LONG_RELATIVE),
;    0070  1  TQ$Internal_Error  : ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    0071  1 %SBTTL 'Included Macros and Literals'
;    0072  1
;    0073  1  $DS_DSADef;    ! Define the DSA bits
;    0074  1  $CRD_Literals;    ! Define the CRD literals
```

```
;    0075  1 %SBTTL 'Psect Definitions'
;    0076  1
;    0077  1 PSECT
;    0078  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0079  1
;    0080  1 PSECT
;    0081  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0082  1
;    0083  1 PSECT
;    0084  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0085  1
;    0086  1 PSECT
;    0087  1     GLOBAL  - Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0088  1 %SBTTL 'Module-Global Static Storage'
;    0089  1
;    0090  1  ModNam ('CRDTQACT');  ! Module name for ErrSup macro
```

```
;    0091  1 %SBTTL 'TQ$Init_TQ Routine'
;    0092  1
;    0093  1 GLOBAL ROUTINE TQ$Init_TQ =
;    0094  1
;    0095  1 !++
;    0096  1 ! Functional Description:
;    0097  1 !
;    0098  1 ! Describe function of routine
;    0099  1 !
;    0100  1 ! Formal Input Parameters:
;    0101  1 !
;    0102  1 ! None
;    0103  1 !
;    0104  1 ! Formal Output Parameters:
;    0105  1 !
;    0106  1 ! None
;    0107  1 !
;    0108  1 ! Side Effects:
;    0109  1 !
;    0110  1 ! None
;    0111  1 !
;    0112  1 ! Completion Codes:
;    0113  1 !
;    0114  1 ! None
;    0115  1 !--
```

```
;    0116  2 BEGIN      ! Routine TQ$Init_TQ
;    0117  2  BIND
;    0118  2   Test_Queue_Table = (.MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr]) : VECTOR [, LONG];
;    0119  2
;    0120  2  REGISTER
;    0121  2   DDB_Ptr,
;    0122  2   TQT_Index;
;    0123  2
;    0124  2  IF (.CRD$GB_CRD_Debug) THEN
;    0125  2   INCR TQ_Index FROM 0 TO (.MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb_Entries] - 1) DO
;    0126  3    BEGIN
; P  0127  3    $CRD_Print ($ASCIC ('*** Test_Queue_Table [!UL] = !XL!/'),
;    0128  3     .TQ_Index, .Test_Queue_Table [.TQ_Index]);
;    0129  2    END;
;    0130  2
;    0131  2  !+
;    0132  2  ! Start at entry #0 in the Test Queue Table
;    0133  2  !-
;    0134  2
;    0135  2  CRD$GL_TQ_Current_TQ_Entry = 0;
;    0136  2
;    0137  2  !+
;    0138  2  ! Initialize all the status bits in all the DDBs in the Test Queue Table.
;    0139  2  !-
;    0140  2
;    0141  2  TQT_Index = 0;
;    0142  2
;    0143  2  WHILE ((DDB_Ptr = .Test_Queue_Table [.TQT_Index]) NEQ 0) DO
;    0144  3   BEGIN
;    0145  3   CRD$Init_DDB_Status (.DDB_Ptr);
;    0146  3   TQT_Index = .TQT_Index + 1;
;    0147  2   END;
;    0148  2
;    0149  3  RETURN (CRD$K_Repeat_Turing)
;    0150  1 END;      ! Routine TQ$Init_TQ


          .TITLE  CRDTQACT *** CRDTQACT Module
          .IDENT  \01-00\

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

   54  43  41  51  54  44  52  43  08  00000 P.AAA:  .ASCII  <8>\CRDTQACT\          ;
65  75  65  75  51  5F  74  73  65  54  20  2A  2A  2A  22  00009 P.AAB:  .ASCII  \ *** Test_Queue_Table [!UL] = !XL!/\       ;
20  3D  20  5D  4C  55  21  5B  20  65  6C  62  61  54  5F  00018             ;
     2F  21  4C  58  21  00027             ;

          DSA$AL_APTMAIL=     65024
          DSA$GL_FLAGS=       65024
          DSA$GL_APTCOM=      65028
          DSA$GL_PASSES=      65032
          DSA$GL_UNITS=       65036
          DSA$GL_SECTNO=      65040
          DSA$GL_SID-       65044
          DSA$GL_MSGTYP=      65088
```

```
          DSA$GL_ERRNO=       65092
          DSA$GL_EVENT=       65096
          DSA$GL_SUBTNO=      65100
          DSA$GL_TESTNO=      65104
          DSA$GL_PASSNO=      65108
          DSA$GL_DEVLEN=      65112
          DSA$GL_DEVNAM=      65116
          DSA$GL_MSGPTR=      65128
          $MODULE=       P.AAA
           .EXTRN  TQ$INIT_TQ, TQ$GET_DDB
           .EXTRN  TQ$CHANGE_TABLE
           .EXTRN  TQ$DONE_TEST_QUEUE
           .EXTRN  TQ$INTERNAL_ERROR
           .EXTRN  MEN$GA_TEST_QUEUE
           .EXTRN  CRD$GB_CRD_DEBUG
           .EXTRN  CRD$PRINT, CRD$GL_TQ_CURRENT_TQ_ENTRY
           .EXTRN  CRD$INIT_DDB_STATUS

           .PSECT  CODE,NOWRT,  SHR,  PIC,2

             001C 00000     .ENTRY  TQ$INIT_TQ, Save R2,R3,R4        ; 0093
          54 00000000G   EF  D0 00002    MOVL    MEN$GA_TEST_QUEUE+4, R4        ; 0118
          26 00000000G   EF  E9 00009    BLBC    CRD$GB_CRD_DEBUG, 3$          ; 0124
          53 00000000G   EF  D0 00010    MOVL    MEN$GA_TEST_QUEUE, R3          ; 0125
          52       01  CE 00017    MNEGL   #1, TQ_INDEX          ;
          16  11 0001A    BRB     2$            ;
        6442  DD 0001C 1$:   PUSHL   (R4)[TQ_INDEX]          ; 0128
          52  DD 0001F    PUSHL   TQ_INDEX          ;
       00000000'  EF  9F 00021    PUSHAB  P.AAB          ;
          01  DD 00027    PUSHL   #1            ;
          1A  DD 00029    PUSHL   #26           ;
       00000000G  FF       05  FB 0002B    CALLS   #5, @CRD$PRINT          ;
          E6       52       53  F2 00032 2$:   AOBLSS  R3, TQ_INDEX, 1$          ; 0125
       00000000G  EF  D4 00036 3$:   CLRL    CRD$GL_TQ_CURRENT_TQ_ENTRY        ; 0135
          53  D4 0003C    CLRL    TQT_INDEX           ; 0141
          0B  11 0003E    BRB     5$            ; 0143
          52  DD 00040 4$:   PUSHL   DDB_PTR           ; 0145
       00000000G  EF       01  FB 00042    CALLS   #1, CRD$INIT_DDB_STATUS         ;
          53  D6 00049    INCL    TQT_INDEX           ; 0146
          52    6443  D0 0004B 5$:   MOVL    (R4)[TQT_INDEX], DDB_PTR        ; 0143
          EF  12 0004F    BNEQ    4$            ;
          50       02  D0 00051    MOVL    #2, R0           ; 0149
        04 00054    RET                 ; 0150

; Routine Size:  85 bytes,    Routine Base:  CODE + 0000
```

```
;    0151  1 %SBTTL 'TQ$Get_DDB Routine'
;    0152  1
;    0153  1 GLOBAL ROUTINE TQ$Get_DDB =
;    0154  1
;    0155  1 !++
;    0156  1 ! Functional Description:
;    0157  1 !
;    0158  1 ! Describe function of routine
;    0159  1 !
;    0160  1 ! Formal Input Parameters:
;    0161  1 !
;    0162  1 ! None
;    0163  1 !
;    0164  1 ! Formal Output Parameters:
;    0165  1 !
;    0166  1 ! None
;    0167  1 !
;    0168  1 ! Side Effects:
;    0169  1 !
;    0170  1 ! None
;    0171  1 !
;    0172  1 ! Completion Codes:
;    0173  1 !
;    0174  1 ! None
;    0175  1 .--
```

```
;   0176  2 BEGIN     ! Routine TQ$Get_DDB
;   0177  2  BIND
;   0178  2   Test_Queue_Table = (.MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr]) : VECTOR [, LONG];
;   0179  2
;   0180  2  IF (.Test_Queue_Table [.CRD$GL_TQ_Current_TQ_Entry] EQL 0) THEN
;   0181  3   BEGIN
;   0182  3   CRD$GL_TQ_Current_State = TQ$K_State_Done_Test_Queue;
;   0183  3   END
;   0184  2  ELSE
;   0185  3   BEGIN
;   0186  3   IF (.CRD$GB_CRD_Debug) THEN
;   0187  4    BEGIN
;   0188  4    MAP
;   0189  4     CRD$GA_HS_Current_DDB_Ptr : REF Device_Data_Block_Structure;
;   0190  4
;   0191  4    IF (.CRD$GA_HS_Current_DDB_Ptr NEQ 0) THEN
; P 0192  4     $CRD_Print ($ASCIC ('*** Get_DDB: The prev DDB = !XL, status = !XL!/'),
;   0193  5      .CRD$GA_HS_Current_DDB_Ptr, .CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status])
;   0194  4    ELSE
; P 0195  4     $CRD_Print ($ASCIC ('*** Get_DDB: The prev DDB = !XL!/'),
;   0196  4      .CRD$GA_HS_Current_DDB_Ptr);
;   0197  3    END;
;   0198  3
;   0199  3   CRD$GA_HS_Current_DDB_Ptr  = .Test_Queue_Table [.CRD$GL_TQ_Current_TQ_Entry];
;   0200  3   CRD$GL_TQ_Current_TQ_Entry = .CRD$GL_TQ_Current_TQ_Entry + 1;
;   0201  3
;   0202  3   IF (.CRD$GB_CRD_Debug) THEN
;   0203  4    BEGIN
;   0204  4    MAP
;   0205  4     CRD$GA_HS_Current_DDB_Ptr : REF Device_Data_Block_Structure;
;   0206  4
; P 0207  4    $CRD_Print ($ASCIC ('*** Get_DDB: The next DDB = !XL, status = !XL!/'),
;   0208  4     .CRD$GA_HS_Current_DDB_Ptr, .CRD$GA_HS_Current_DDB_Ptr [CRD$K_DDB_Status]);
;   0209  3    END;
;   0210  2   END;
;   0211  2
;   0212  3  RETURN (CRD$K_Repeat_Turing)
;   0213  1 END;     ! Routine TQ$Get_DDB


          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

54  20  3A  42  44  44  5F  74  65  47  20  2A  2A  2A  2F  0002C P.AAC:  .ASCII  \/*** Get_DDB: The prev DDB = !XL, status\  ;
21  20  3D  20  42  44  44  20  76  65  72  70  20  65  68  0003B             ;
      73  75  74  61  74  73  20  2C  4C  58  0004A                 ;
        2F  21  4C  58  21  20  3D  20  00054     .ASCII  \ = !XL!/\          ;
54  20  3A  42  44  44  5F  74  65  47  20  2A  2A  2A  21  0005C P.AAD:  .ASCII  \!*** Get_DDB: The prev DDB = !XL!/\       ;
21  20  3D  20  42  44  44  20  76  65  72  70  20  65  68  0006B             ;
          2F  21  4C  58  0007A                 ;
54  20  3A  42  44  44  5F  74  65  47  20  2A  2A  2A  2F  0007E P.AAE:  .ASCII  \/*** Get_DDB: The next DDB = !XL, status\  ;
21  20  3D  20  42  44  44  20  74  78  65  6E  20  65  68  0008D             ;
      73  75  74  61  74  73  20  2C  4C  58  0009C                 ;
        2F  21  4C  58  21  20  3D  20  000A6     .ASCII  \ = !XL./\          ;
```

```
                .EXTRN  CRD$GL_TQ_CURRENT_STATE
                .EXTRN  CRD$GA_HS_CURRENT_DDB_PTR

                .PSECT  CODE,NOWRT,  SHR,  PIC,2

                  0004 00000     .ENTRY  TQ$GET_DDB, Save R2          ; 0153
                52 00000000G   EF  D0 00002     MOVL    MEN$GA_TEST_QUEUE+4, R2          ; 0178
                50 00000000G   EF  D0 00009     MOVL    CRD$GL_TQ_CURRENT_TQ_ENTRY, R0          ; 0180
              6240  D5 00010     TSTL    (R2)[R0]          ;
                09  12 00013     BNEQ    1$              ;
             00000000G  EF      07  D0 00015     MOVL    #7, CRD$GL_TQ_CURRENT_STATE          ; 0182
                74  11 0001C     BRB     4$              ; 0180
                34 00000000G   EF  E9 0001E 1$:   BLBC    CRD$GB_CRD_DEBUG, 3$          ; 0186
                50 00000000G   EF  D0 00025     MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0          ; 0191
                18  13 0002C     BEQL    2$              ;
            14  A0  DD 0002E     PUSHL   20(R0)            ; 0193
                50  DD 00031     PUSHL   R0              ;
             00000000'  EF  9F 00033     PUSHAB  P.AAC           ;
                01  DD 00039     PUSHL   #1              ;
                1A  DD 0003B     PUSHL   #26             ;
             00000000G  FF      05  FB 0003D     CALLS   #5, @CRD$PRINT          ;
                13  11 00044     BRB     3$              ;
                50  DD 00046 2$:   PUSHL   R0              ; 0196
             00000000'  EF  9F 00048     PUSHAB  P.AAD           ;
                01  DD 0004E     PUSHL   #1              ;
                1A  DD 00050     PUSHL   #26             ;
             00000000G  FF      04  FB 00052     CALLS   #4, @CRD$PRINT          ;
                50 00000000G   EF  D0 00059 3$:   MOVL    CRD$GL_TQ_CURRENT_TQ_ENTRY, R0          ; 0199
             00000000G  EF    6240  D0 00060     MOVL    (R2)[R0], CRD$GA_HS_CURRENT_DDB_PTR          ;
             00000000G  EF  D6 00068     INCL    CRD$GL_TQ_CURRENT_TQ_ENTRY          ; 0200
                1D 00000000G   EF  E9 0006E     BLBC    CRD$GB_CRD_DEBUG, 4$          ; 0202
                50 00000000G   EF  D0 00075     MOVL    CRD$GA_HS_CURRENT_DDB_PTR, R0          ; 0208
            14  A0  DD 0007C     PUSHL   20(R0)            ;
                50  DD 0007F     PUSHL   R0              ;
             00000000'  EF  9F 00081     PUSHAB  P.AAE           ;
                01  DD 00087     PUSHL   #1              ;
                1A  DD 00089     PUSHL   #26             ;
             00000000G  FF      05  FB 0008B     CALLS   #5, @CRD$PRINT          ;
                50      02  D0 00092 4$:   MOVL    #2, R0          ; 021^
                 04 00095     RET                ; 0213

; Routine Size:  150 bytes,    Routine Base:  CODE + 0055
```

```
;    0214  1 %SBTTL 'TQ$Change_Table Routine'
;    0215  1
;    0216  1 GLOBAL ROUTINE TQ$Change_Table =
;    0217  1
;    0218  1 !++
;    0219  1 ! Functional Description:
;    0220  1 !
;    0221  1 ! Describe function of routine
;    0222  1 !
;    0223  1 ! Formal Input Parameters:
;    0224  1 !
;    0225  1 ! None
;    0226  1 !
;    0227  1 ! Formal Output Parameters:
;    0228  1 !
;    0229  1 ! None
;    0230  1 !
;    0231  1 ! Side Effects:
;    0232  1 !
;    0233  1 ! None
;    0234  1 !
;    0235  1 ! Completion Codes:
;    0236  1 !
;    0237  1 ! None
;    0238  1 --
```

```
;    0239  2 BEGIN     ! Routine TQ$Change_Table
;    0240  2  CRD$GB_Current_State_Table = MEN$K_HS_State_Table;
;    0241  2       ! Change to the Hardware Support State Table now.
;    0242  3  RETURN (CRD$K_Repeat_Turing)
;    0243  1 END;     ! Routine TQ$Change_Table



            .EXTRN  CRD$GB_CURRENT_STATE_TABLE

              0000 00000     .ENTRY  TQ$CHANGE_TABLE, Save nothing          ; 0216
        00000000G  EF       02  90 00002     MOVB    #2, CRD$GB_CURRENT_STATE_TABLE       ; 0240
            50       02  D0 00009     MOVL    #2, R0                ; 0242
           04 0000C     RET                 ; 0243

; Routine Size:  13 bytes,    Routine Base:  CODE + 00EB
```

```
;    0244  1 %SBTTL 'TQ$Done_Test_Queue Routine'
;    0245  1
;    0246  1 GLOBAL ROUTINE TQ$Done_Test_Queue =
;    0247  1
;    0248  1 !++
;    0249  1 ! Functional Description:
;    0250  1 !
;    0251  1 ! Describe function of routine
;    0252  1 !
;    0253  1 ! Formal Input Parameters:
;    0254  1 !
;    0255  1 ! None
;    0256  1 !
;    0257  1 ! Formal Output Parameters:
;    0258  1 !
;    0259  1 ! None
;    0260  1 !
;    0261  1 ! Side Effects:
;    0262  1 !
;    0263  1 ! None
;    0264  1 !
;    0265  1 ! Completion Codes:
;    0266  1 !
;    0267  1 ! None
;    0268  1 !
```

```
;    0269  2 BEGIN     ! Routine TQ$Done_Test_Queue
;    0270  2  !+
;    0271  2  ! Re-initialize TQ:
;    0272  2  !-
;    0273  2
;    0274  2  MEN$FncTst_CmplErr_Status (); ! Print an ending message - how many errors, etc.
;    0275  2
;    0276  2  CRD$GL_TQ_Current_State = TQ$K_State_Init_TQ;
;    0277  2      ! Reset the Test Queue state number
;    0278  2  CRD$GB_Current_State_Table = MEN$K_MM_State_Table;
;    0279  2      ! Change to the Main Menu state table
;    0280  3  RETURN (CRD$K_Repeat_Turing)
;    0281  1 END;     ! Routine TQ$Done_Test_Queue



          .EXTRN  MEN$FNCTST_CMPLERR_STATUS

            0000 00000     .ENTRY  TQ$DONE_TEST_QUEUE, Save nothing        ; 0246
      00000000G  EF       00  FB 00002     CALLS   #0, MEN$FNCTST_CMPLERR_STATUS        ; 0274
      00000000G  EF       03  D0 00009     MOVL    #3, CRD$GL_TQ_CURRENT_STATE          ; 0276
      00000000G  EF  94 00010     CLRB    CRD$GB_CURRENT_STATE_TABLE          ; 0278
          50       02  D0 00016     MOVL    #2, R0              ; 0280
       04 00019     RET                 ; 0281

; Routine Size:  26 bytes,    Routine Base:  CODE + 00F8
```

```
;    0282  1 %SBTTL 'TQ$Internal_Error Routine'
;    0283  1
;    0284  1 GLOBAL ROUTINE TQ$Internal_Error (TypeCode, Length, Address) =
;    0285  1
;    0286  1 !++
;    0287  1 ! Functional Description:
;    0288  1 !
;    0289  1 ! Describe function of routine
;    0290  1 !
;    0291  1 ! Formal Input Parameters:
;    0292  1 !
;    0293  1 ! None
;    0294  1 !
;    0295  1 ! Formal Output Parameters:
;    0296  1 !
;    0297  1 ! None
;    0298  1 !
;    0299  1 ! Side Effects:
;    0300  1 !
;    0301  1 ! None
;    0302  1 !
;    0303  1 ! Completion Codes:
;    0304  1 !
;    0305  1 ! None
;    0306  1 .
```

```
;   0307  2 BEGIN     ! Routine TQ$Internal_Error
;   0308  2  MEN$Menu_Test_Internal_Error (CRD$K_TQ_Internal_Error, .TypeCode, .Length, .Address);
;   0309  2  CRD$Stop (AUTO$K_Exit_Internal_Error);
;   0310  3  RETURN (CRD$K_Return_Failure)
;   0311  1 END;     ! Routine TQ$Internal_Error



          .EXTRN  MEN$MENU_TEST_INTERNAL_ERROR
          .EXTRN  CRD$STOP

            0000 00000     .ENTRY  TQ$INTERNAL_ERROR, Save nothing       ; 0284
          7E 08    AC  7D 00002     MOVQ    LENGTH, -(SP)           ; 0308
       04 AC  DD 00006     PUSHL   TYPECODE          ;
          7E       07  CE 00009     MNEGL   #7, -(SP)         ;
       00000000G  EF       04  FB 0000C     CALLS   #4, MEN$MENU_TEST_INTERNAL_ERROR       ;
          01  DD 00013     PUSHL   #1              ; 0309
       00000000G  EF       01  FB 00015     CALLS   #1, CRD$STOP          ;
          50  D4 0001C     CLRL    R0              ; 0310
           04 0001E     RET                 ; 0311

; Routine Size:  31 bytes,    Routine Base:  CODE + 0112
```

```
;   0312  1 END     ! End of CRDTQACT module
;   0313  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          174  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          305  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- -------- -------
$ASCIC    Macro    Lib02   127     192     195     207
$CRD_LITERALS   Macro    Lib03    74
$CRD_PRINT    Macro    Lib03   127     192     195     207
$DS_DSADEF    Macro    Lib02    73
$DS_TYPEDEF    Macro    Lib02   128     193     196     208
$EQULST    Macro    Lib04    74     128     193     196     208
$ER    Comptime    90
$MODULE    Bind        90
ADDRESS    RoutForm   284   308.
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     74
AUTO$K_EXIT_CONTROL_C  Literal     74
AUTO$K_EXIT_DUMMY_2  Literal     74
AUTO$K_EXIT_DUMMY_3  Literal     74
AUTO$K_EXIT_ERRORS_COMPLETE Literal     74
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     74
AUTO$K_EXIT_INTERNAL_ERROR Literal     74   309
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     74
AUTO$K_EXIT_LAST_REASON  Literal     74     74
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     74
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     74
AUTO$K_EXIT_NOERRORS_PREP Literal     74
CRD$GA_HS_CURRENT_DDB_PTR External Lib03   189m    191.    193.    193a.   196.    199=
    Map         205   208.    208a.
CRD$GB_CRD_DEBUG  External Lib03   124.    186.    202.
CRD$GB_CURRENT_STATE_TABLE External Lib03   240=    278=
CRD$GL_TQ_CURRENT_STATE  External Lib03   182=    276=
CRD$GL_TQ_CURRENT_TQ_ENTRY External Lib03   135=    180.    199.    200=    200.
CRD$INIT_DDB_STATUS  ExtRout  Lib03   145c
CRD$K_ACTION_EQL_DO_NOTHING Literal     74
CRD$K_ACTION_RANGE_ERROR Literal     74
CRD$K_ADVANCE_DIAGNOSTIC Literal     74
CRD$K_ADVANCE_GENERAL_COM Literal     74
CRD$K_ADVANCE_STATE  Literal     74
CRD$K_ALLOCATION_ERROR  Literal     74
```

Symbol	Type	Defined	Referenced ...
CRD$K_ASSIGN_PTABLE_PTRS	Literal	74	
CRD$K_AUTOSIZER_ERROR	Literal	74	
CRD$K_AUTOSIZER_SET_FLAGS	Literal	74	
CRD$K_BAD_ACTION_NUMBER	Literal	74	
CRD$K_BAD_TYPECODE_ERROR	Literal	74	
CRD$K_BASE_SYSTEM_IDC	Literal	74	
CRD$K_BASE_SYSTEM_LESI	Literal	74	
CRD$K_BASE_SYSTEM_NULL	Literal	74	
CRD$K_BA_ _SYSTEM_UDA_0	Literal	74	
CRD$K_BASE_SYSTEM_UDA_1	Literal	74	
CRD$K_BASE_SYSTEM_UDA_2	Literal	74	
CRD$K_BASE_SYSTEM_UDA_3	Literal	74	
CRD$K_CRD_INITIALIZATION	Literal	74	
CRD$K_CREATE_HST	Literal	74	
CRD$K_DATA_LENGTH_ERROR	Literal	74	
CRD$K_DDB_AUTO_SUPPORT_ENTRY	Literal	74	
CRD$K_DDB_BLINK	Literal	74	
CRD$K_DDB_FLINK	Literal	74	
CRD$K_DDB_LAST_ENTRY	Literal	74	
CRD$K_DDB_MEMORY_SIZE	Literal	74	
CRD$K_DDB_MENU_DEVICE_NUMB	Literal	74	
CRD$K_DDB_MENU_SUPPORT_ENTRY	Literal	74	
CRD$K_DDB_MENU_SUPPORT_SIZE	Literal	74	
CRD$K_DDB_NUMB_SUPPORT_TABLES	Literal	74	
CRD$K_DDB_PTABLE_ENTRY	Literal	74	
CRD$K_DDB_STATUS	Literal	74	193 208
CRD$K_DDB_SUPBLK_BLOCK_PTR	Literal	74	
CRD$K_DEVICE_NAME	Literal	74	
CRD$K_DIAGNOSTIC_ERROR	Literal	74	
CRD$K_DIAGNOSTIC_NAME	Literal	74	
CRD$K_DONE_GENERAL_COMS	Literal	74	
CRD$K_DO_NOTHING	Literal	74	
CRD$K_DUMMY_10	Literal	74	
CRD$K_DUMMY_11	Literal	74	
CRD$K_DUMMY_12	Literal	74	
CRD$K_DUMMY_13	Literal	74	
CRD$K_DUMMY_3	Literal	74	
CRD$K_DUMMY_4	Literal	74	
CRD$K_DUMMY_5	Literal	74	
CRD$K_DUMMY_6	Literal	74	
CRD$K_DUMMY_7	Literal	74	
CRD$K_DUMMY_8	Literal	74	
CRD$K_DUMMY_9	Literal	74	
CRD$K_EXIT_CRD	Literal	74	
CRD$K_HOOK_POINT	Literal	74	
CRD$K_HS_INTERNAL_ERROR	Literal	74	
CRD$K_IDC_EVDLB	Literal	74	
CRD$K_IDC_EVDLC	Literal	74	
CRD$K_IDC_EVDLD	Literal	74	
CRD$K_IDC_EVRAD_0	Literal	74	
CRD$K_IDC_EVRAD_1	Literal	74	
CRD$K_IDC_LAST_DIAGNOSTIC	Literal	74	
CRD$K_INTERNAL_ERROR	Literal	74	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_LAST_ACTION_ROUTINE Literal     74
CRD$K_LAST_ENTRY  Literal     74
CRD$K_LAST_FUNCTION  Literal     74
CRD$K_LAST_HOOK_POINT  Literal     74
CRD$K_LAST_INTERNAL_ERROR Literal     74
CRD$K_LAST_TYPE   Literal     74
CRD$K_LESI_ENWCA  Literal     74
CRD$K_LESI_EVRMB_0  Literal     74
CRD$K_LESI_EVRMB_1  Literal     74
CRD$K_LESI_LAST_DIAGNOSTIC Literal     74
CRD$K_LEVEL_4_PASSED  Literal     74
CRD$K_LOAD_AUTOSIZER  Literal     74
CRD$K_LOAD_ERROR  Literal     74
CRD$K_LOAD_GENERAL_COM  Literal     74
CRD$K_LOAD_LOAD_COM  Literal     74
CRD$K_LOAD_SELECT_COM  Literal     74
CRD$K_LOAD_SET_EVENT_COM Literal     74
CRD$K_LOAD_SET_FLAGS_COM Literal     74
CRD$K_LOAD_START_COM  Literal     74
CRD$K_LOAD_SUP_FILE  Literal     74
CRD$K_MM_INTERNAL_ERROR  Literal     74
CRD$K_NEW_LINE   Literal     74
CRD$K_PREPARATION_ERROR  Literal     74
CRD$K_PREPARATION_ERROR_TEXT Literal     74
CRD$K_REPEAT_TURING  Literal  Lib03   149     212     242     280
CRD$K_RETURN_FAILURE  Literal  Lib03   310
CRD$K_RUNTIME   Literal     74
CRD$K_SAME_LINE   Literal     74
CRD$K_SET_EVENT_ARGS  Literal     74
CRD$K_SET_FLAGS_ARGS  Literal     74
CRD$K_SET_UP_DIAGNOSTIC  Literal     74
CRD$K_START   Literal     74
CRD$K_START_AUTOSIZER  Literal     74
CRD$K_START_ERROR  Literal     74
CRD$K_START_QUALIFIERS  Literal     74
CRD$K_STAR_LINE   Literal     74
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal     74
CRD$K_STATE_ADVANCE_GENERAL_COM Literal     74
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal     74
CRD$K_STATE_AUTOSIZER_ERROR Literal     74
CRD$K_STATE_AUTOSIZER_RUNNING Literal     74
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal     74
CRD$K_STATE_DIAGNOSTIC_ERROR Literal     74
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal     74
CRD$K_STATE_DONE_DIAGNOSTICS Literal     74
CRD$K_STATE_DONE_GENERAL_COMS Literal     74
CRD$K_STATE_DUMMY_1  Literal     74
CRD$K_STATE_DUMMY_10  Literal     74
CRD$K_STATE_DUMMY_12  Literal     74
CRD$K_STATE_DUMMY_13  Literal     74
CRD$K_STATE_DUMMY_2  Literal     74
CRD$K_STATE_DUMMY_3  Literal     74
CRD$K_STATE_DUMMY_4  Literal     74
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_DUMMY_6  Literal     74
CRD$K_STATE_DUMMY_7  Literal     74
CRD$K_STATE_DUMMY_8  Literal     74
CRD$K_STATE_EXIT_CRD  Literal     74
CRD$K_STATE_INTERNAL_ERROR Literal     74
CRD$K_STATE_LAST_STATE  Literal     74
CRD$K_STATE_LEVEL_4_PASSED Literal     74
CRD$K_STATE_LOAD_AUTOSIZER Literal     74
CRD$K_STATE_LOAD_ERROR  Literal     74
CRD$K_STATE_LOAD_GENERAL_COM Literal     74
CRD$K_STATE_LOAD_LOAD_COM Literal     74
CRD$K_STATE_LOAD_SELECT_COM Literal     74
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     74
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     74
CRD$K_STATE_LOAD_START_COM Literal     74
CRD$K_STATE_PREPARATION_ERROR Literal     74
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     74
CRD$K_STATE_START  Literal     74
CRD$K_STATE_START_AUTOSIZER Literal     74
CRD$K_STATE_START_ERROR  Literal     74
CRD$K_TEST_START_TEXT  Literal     74
CRD$K_TQ_INTERNAL_ERROR  Literal     74   308
CRD$K_TYPECODE   Literal     74
CRD$K_UDA_0_EVDLB  Literal     74
CRD$K_UDA_0_EVDLC  Literal     74
CRD$K_UDA_0_EVDLD  Literal     74
CRD$K_UDA_0_EVRLA_0  Literal     74
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     74
CRD$K_UDA_1_EVDLB  Literal     74
CRD$K_UDA_1_EVDLC  Literal     74
CRD$K_UDA_1_EVDLD  Literal     74
CRD$K_UDA_1_EVRLA_0  Literal     74
CRD$K_UDA_1_EVRLA_1  Literal     74
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     74
CRD$K_UDA_2_EVDLB  Literal     74
CRD$K_UDA_2_EVDLC  Literal     74
CRD$K_UDA_2_EVDLD  Literal     74
CRD$K_UDA_2_EVRLA_0  Literal     74
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     74
CRD$K_UDA_3_EVDLB  Literal     74
CRD$K_UDA_3_EVDLC  Literal     74
CRD$K_UDA_3_EVDLD  Literal     74
CRD$K_UDA_3_EVRLA_0  Literal     74
CRD$K_UDA_3_EVRLA_1  Literal     74
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     74
CRD$PRINT   External Lib03   128ac   193ac   196ac   208ac
CRD$STOP   ExtRout  Lib03   309c
DDB_PTR    Register   121   143=    145.
DEVICE_DATA_BLOCK_STRUCTURE Structur Lib03   189     205
DS$K_PRINTB   Literal    128
    Literal    193
    Literal    196
    Literal    208
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DS$K_PRINTF   Literal    128   128
    Literal     193   193
    Literal     196   196
    Literal     208   208
DS$K_PRINTI   Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_PRINTX   Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ABORT_PROGRAM  Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ABORT_TEST  Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_COMMAND_ERR  Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_COMMAND_OUT  Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_CRD_AUTOTEST  Literal    128   128
    Literal     193   193
    Literal     196   196
    Literal     208   208
DS$K_TYPE_DS_PROMPT  Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_DS_START  Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ERRDEV  Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ERRHARD  Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ERROR_BODY  Literal    128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ERROR_END  Literal    128
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ERRPREP  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ERRSOFT  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ERRSUP  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ERRSYS  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_ERR_HALT  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_EXCEPTION  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_EXCEPTION_HEAD Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_FIRST_PASS  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_GENERAL  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_GENERAL_ERROR  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_NO_TESTS  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_PARAM_ERROR  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_PROGRAM_END  Literal     128
    Literal     193
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     196
    Literal     208
DS$K_TYPE_PROGRAM_INFO  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_PROGRAM_START  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_QIO_INVADP  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_QIO_NODRIVER  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_QIO_WRONGVER  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_SCRIPT_ECHO  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_SCRIPT_PNF  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_SCRIPT_PROMPT  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_SCRIPT_SKIP  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_SEQUENCE_ERROR Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_START_ERR  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_START_LIST  Literal     128
    Literal     193
    Literal     196
    Literal     208
DS$K_TYPE_SUMMARY  Literal     128
    Literal     193
    Literal     196
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    208
DS$K_TYPE_USER_PROMPT  Literal    128
    Literal    193
    Literal    196
    Literal    208
DSA$AL_APTMAIL   Bind         73    73
DSA$GL_APTCOM   Bind         73
DSA$GL_DEVLEN   Bind         73
DSA$GL_DEVNAM   Bind         73
DSA$GL_ERRNO   Bind         73
DSA$GL_EVENT   Bind         73
DSA$GL_FLAGS   Bind         73
DSA$GL_MSGPTR   Bind         73
DSA$GL_MSGTYP   Bind         73
DSA$GL_PASSES   Bind         73
DSA$GL_PASSNO   Bind         73
DSA$GL_SECTNO   Bind         73
DSA$GL_SID   Bind         73
DSA$GL_SUBTNO   Bind         73
DSA$GL_TESTNO   Bind         73
DSA$GL_UNITS   Bind         73
GET1ST_    Macro    Lib04    74     128     193     196     208
GET2ND_    Unbound           74     128     193     196     208
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal     74
HS$K_ACTION_ADVANCE_GENERAL_COM Literal     74
HS$K_ACTION_ADVANCE_STATE Literal     74
HS$K_ACTION_CHANGE_TABLE Literal     74
HS$K_ACTION_DIAGNOSTIC_ERROR Literal     74
HS$K_ACTION_DONE_GENERAL_COMS Literal     74
HS$K_ACTION_DO_NOTHING  Literal     74
HS$K_ACTION_DUMMY_3  Literal     74
HS$K_ACTION_DUMMY_4  Literal     74
HS$K_ACTION_DUMMY_5  Literal     74
HS$K_ACTION_DUMMY_6  Literal     74
HS$K_ACTION_INIT_HS  Literal     74
HS$K_ACTION_INTERNAL_ERROR Literal     74
HS$K_ACTION_LAST_ACTION  Literal     74
HS$K_ACTION_LOAD_ERROR  Literal     74
HS$K_ACTION_LOAD_GENERAL_COM Literal     74
HS$K_ACTION_LOAD_LOAD_COM Literal     74
HS$K_ACTION_LOAD_SELECT_COM Literal     74
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     74
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     74
HS$K_ACTION_LOAD_START_COM Literal     74
HS$K_ACTION_PREPARATION_ERROR Literal     74
HS$K_ACTION_START_ERROR  Literal     74
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     74
HS$K_STATE_ADVANCE_GENERAL_COM Literal     74
HS$K_STATE_CHANGE_TABLE  Literal     74
HS$K_STATE_DIAGNOSTIC_ERROR Literal     74
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     74
HS$K_STATE_DONE_GENERAL_COMS Literal     74
HS$K_STATE_DUMMY_1  Literal     74
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_STATE_DUMMY_2  Literal     74
HS$K_STATE_DUMMY_3  Literal     74
HS$K_STATE_DUMMY_4  Literal     74
HS$K_STATE_DUMMY_6  Literal     74
HS$K_STATE_INIT_HS  Literal     74
HS$K_STATE_INTERNAL_ERROR Literal     74
HS$K_STATE_LAST_STATE  Literal     74
HS$K_STATE_LOAD_ERROR  Literal     74
HS$K_STATE_LOAD_GENERAL_COM Literal     74
HS$K_STATE_LOAD_LOAD_COM Literal     74
HS$K_STATE_LOAD_SELECT_COM Literal     74
HS$K_STATE_LOAD_SET_EVENT_COM Literal     74
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     74
HS$K_STATE_LOAD_START_COM Literal     74
HS$K_STATE_PREPARATION_ERROR Literal     74
HS$K_STATE_START_ERROR  Literal     74
LENGTH    RoutForm   284   308.
MEN$FNCTST_CMPLERR_STATUS ExtRout  Lib03   274c
MEN$GA_TEST_QUEUE  External Lib03   118.    125.    178.
MEN$K_HS_STATE_TABLE  Literal     74   240
MEN$K_MM_STATE_TABLE  Literal     74   278
MEN$K_TQ_STATE_TABLE  Literal     74
MEN$MENU_TEST_INTERNAL_ERROR ExtRout  Lib03   308c
MEN$V_TQ_TABLE_NUMB_ENTRIES Field    Lib03   125.
MEN$V_TQ_TABLE_PTR  Field    Lib03   118.    178.
MENU$K_EXIT_AUTOSIZER_ERROR Literal     74
MENU$K_EXIT_INTERNAL_ERROR Literal     74
MENU$K_EXIT_LAST_REASON  Literal     74
MENU$K_EXIT_OPERATOR_ABORT Literal     74
MENU$K_EXIT_OPERATOR_STOP Literal     74
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     74
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     74
MM$K_ACTION_ADVANCE_STATE Literal     74
MM$K_ACTION_AUTOSIZER_ERROR Literal     74
MM$K_ACTION_BUILD_DDBS  Literal     74
MM$K_ACTION_BUILD_HSTS  Literal     74
MM$K_ACTION_CHANGE_TABLE Literal     74
MM$K_ACTION_DONE_INITS  Literal     74
MM$K_ACTION_DO_NOTHING  Literal     74
MM$K_ACTION_DUMMY_3  Literal     74
MM$K_ACTION_DUMMY_4  Literal     74
MM$K_ACTION_DUMMY_5  Literal     74
MM$K_ACTION_DUMMY_6  Literal     74
MM$K_ACTION_DUMMY_7  Literal     74
MM$K_ACTION_DUMMY_8  Literal     74
MM$K_ACTION_INTERNAL_ERROR Literal     74
MM$K_ACTION_LAST_ACTION  Literal     74
MM$K_ACTION_LOAD_AUTOSIZER Literal     74
MM$K_ACTION_MENU_PROMPT  Literal     74
MM$K_ACTION_PRINT_MENU  Literal     74
MM$K_ACTION_PRINT_MENU_HEADING Literal     74
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     74
MM$K_ACTION_START_AUTOSIZER Literal     74
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_STATE_AUTOSIZER_ERROR Literal     74
MM$K_STATE_AUTOSIZER_RUNNING Literal     74
MM$K_STATE_BUILD_DDBS  Literal     74
MM$K_STATE_BUILD_HSTS  Literal     74
MM$K_STATE_CHANGE_TABLE  Literal     74
MM$K_STATE_DONE_INITS  Literal     74
MM$K_STATE_DUMMY_1  Literal     74
MM$K_STATE_DUMMY_2  Literal     74
MM$K_STATE_DUMMY_3  Literal     74
MM$K_STATE_DUMMY_5  Literal     74
MM$K_STATE_DUMMY_7  Literal     74
MM$K_STATE_DUMMY_8  Literal     74
MM$K_STATE_INTERNAL_ERROR Literal     74
MM$K_STATE_LAST_STATE  Literal     74
MM$K_STATE_LOAD_AUTOSIZER Literal     74
MM$K_STATE_MENU_PROMPT  Literal     74
MM$K_STATE_PRINT_MENU  Literal     74
MM$K_STATE_PRINT_MENU_HEADING Literal     74
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     74
MM$K_STATE_START  Literal     74
MM$K_STATE_START_AUTOSIZER Literal     74
MODNAM    Macro    Lib01    90
NUL2ND    Macro    Lib04    74     128     193     196     208
TEST_QUEUE_TABLE  Bind       118   128.    143.
    Bind       178   180.    199.
TQ$CHANGE_TABLE   GlobRout   216 Lib03e     68f
TQ$DONE_TEST_QUEUE  GlobRout   246 Lib03e     69f
TQ$GET_DDB   GlobRout   153 Lib03e     67f
TQ$INIT_TQ   GlobRout    93 Lib03e     66f
TQ$INTERNAL_ERROR  GlobRout   284 Lib03e     70f
TQ$K_ACTION_ADVANCE_STATE Literal     74
TQ$K_ACTION_CHANGE_TABLE Literal     74
TQ$K_ACTION_DONE_TEST_QUEUE Literal     74
TQ$K_ACTION_DO_NOTHING  Literal     74
TQ$K_ACTION_DUMMY_3  Literal     74
TQ$K_ACTION_DUMMY_4  Literal     74
TQ$K_ACTION_DUMMY_5  Literal     74
TQ$K_ACTION_GET_DDB  Literal     74
TQ$K_ACTION_INIT_TQ  Literal     74
TQ$K_ACTION_INTERNAL_ERROR Literal     74
TQ$K_ACTION_LAST_ACTION  Literal     74
TQ$K_STATE_CHANGE_TABLE  Literal     74
TQ$K_STATE_DONE_TEST_QUEUE Literal     74   182
TQ$K_STATE_DUMMY_1  Literal     74
TQ$K_STATE_DUMMY_2  Literal     74
TQ$K_STATE_DUMMY_3  Literal     74
TQ$K_STATE_DUMMY_4  Literal     74
TQ$K_STATE_DUMMY_5  Literal     74
TQ$K_STATE_GET_DDB  Literal     74
TQ$K_STATE_INIT_TQ  Literal     74   276
TQ$K_STATE_INTERNAL_ERROR Literal     74
TQ$K_STATE_LAST_STATE  Literal     74
TQT_INDEX   Register   122   141=    143.    146=    146.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
TQ_INDEX    Local       125    128.
TYPECODE    RoutForm    284    308.



  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDTQACT.B32;1
    2  Module CRDTQACT
   53  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   56  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   59  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   62  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  313  Eludom CRDTQACT


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     a Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923          3          0          94          00:00.2
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486          25         5          62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0       1000          00:04.5




;     COMMAND QUALIFIERS
```

```
; BLISS CRDTQACT.B32/LIST=CRDTQACT.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDTQACT.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  305 code + 174 data bytes
; Run Time:    00:26.8
; Elapsed Time:    00:51.6
; Lines/CPU Min:     700
; Lexemes/CPU-Min: 63340
; Memory Used: 184 pages
; Compilation Complete
```

```
;    0001  0 %TITLE '*** CRDTYPCOD Module'
;    0002  0 MODULE CRDTYPCOD (    ! Print a message saying what a given typecode is.
;    0003  0   IDENT = '01-01'
;    0004  0   ) =
;    0005  0 !++
;    0006  0 ! COPYRIGHT (c) 1980, 1981
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the CRD-Autotest image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! Print a short message saying what a given typecode is used for.
;    0032  0 !
;    0033  0 ! Author:
;    0034  0 !
;    0035  0 ! Jack Stansbury
;    0036  0 !
;    0037  0 ! Creation Date:
;    0038  0 !
;    0039  0 ! April 29, 1982
;    0040  0 !
;    0041  0 ! Version:
;    0042  0 !
;    0043  0 ! 6.7
;    0044  0 !
;    0045  0 ! Modified By:
;    0046  0 !
;    0047  0 ! 01 Jack Stansbury, March 3, 1983, Version ?.?
;    0048  0 !  Added an assert statement for the number of typcodes. If this number changes
;    0049  0 !  from 38, then changes must be made to the CASE statement below.
;    0050  0 !--
```

```
;    0051  0 %SBTTL 'Libraries'
;    0052  1 BEGIN     ! Begin the CRDTYPCOD module
;    0053  1
;    0054  1 LIBRARY
;    0055  1   '$DS';     ! Use Ds.L32 first
;    0056  1
;    0057  1 LIBRARY
;    0058  1   '$Diag';    ! Use Diag.L32 second
;    0059  1
;    0060  1 LIBRARY
;    0061  1   '$CRD';     ! Use CRD.L32 third
;    0062  1
;    0063  1 LIBRARY
;    0064  1   'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0065  1 %SBTTL 'Table of Contents'
;    0066  1
;    0067  1 FORWARD ROUTINE
;    0068  1  CRD$Print_TypeCode_Name  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
;    0069  1      ! Print the message
```

```
;    0070  1 %SBTTL 'Included Macros and Literals'
;    0071  1
;    0072  1  $DS_TypeDef;    ! The typecde mnemonics
```

```
;    0073  1 %SBTTL 'Psect Definitions'
;    0074  1
;    0075  1 PSECT
;    0076  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0077  1
;    0078  1 PSECT
;    0079  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0080  1
;    0081  1 PSECT
;    0082  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0083  1
;    0084  1 PSECT
;    0085  1     GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;   0086  1 %SBTTL 'Module-Global Static Storage'
;   0087  1
;   0088  1  ModNam ('CRDTYPCOD');  ! Module name for ErrSup macro
```

```
;    0089  1 %SBTTL 'CRD$Print_TypeCode_Name Routine'
;    0090  1
;    0091  1 GLOBAL ROUTINE CRD$Print_TypeCode_Name (TypeCode) : NOVALUE =
;    0092  1
;    0093  1 !++
;    0094  1 ! Functional Description:
;    0095  1 !
;    0096  1 ! Print a short message saying what a given typecode is used for.
;    0097  1 !
;    0098  1 ! Formal Input Parameters:
;    0099  1 !
;    0100  1 ! TypeCode: The typecode constant.
;    0101  1 !
;    0102  1 ! Formal Output Parameters:
;    0103  1 !
;    0104  1 ! None
;    0105  1 !
;    0106  1 ! Side Effects:
;    0107  1 !
;    0108  1 ! None
;    0109  1 !
;    0110  1 ! Completion Codes:
;    0111  1 !
;    0112  1 ! None
;    0113  1 !--
```

```
;    0114  2 BEGIN     ! Routine CRD$Print_TypeCode_Name
;    0115  2
;    0116  2  MACRO
; M  0117  2   TypeCode_Case (TypeCode_Name) =
; M  0118  2    [%NAME ('DS$K_Type_', TypeCode_Name)]:
; M  0119  2     $CRD_Print ($ASCIC (TypeCode_Name, '!/'))
;    0120  2    %;
;    0121  2
; P  0122  2   CRD_Assert ((CRD$K_Numb_Typecodes EQL 38),        ![01]
; L  0123  2    'You must make changes to the CASE statement below if you add more typecodes');   ![01]
; %PRINT: ASSERT: The assertion is true.
;    0124  2
;    0125  2   CASE .TypeCode FROM 0 TO (CRD$K_Numb_TypeCodes - 1) OF
;    0126  2    SET
;    0127  2    TypeCode_Case ('General');
;    0128  2    TypeCode_Case ('DS_Prompt');
;    0129  2    TypeCode_Case ('User_Prompt');
;    0130  2    TypeCode_Case ('General_Error');
;    0131  2    TypeCode_Case ('Errsup');
;    0132  2    TypeCode_Case ('Errsys');
;    0133  2    TypeCode_Case ('Errhard');
;    0134  2    TypeCode_Case ('Errsoft');
;    0135  2    TypeCode_Case ('Errdev');
;    0136  2    TypeCode_Case ('Error_Body');
;    0137  2    TypeCode_Case ('Error_End');
;    0138  2    TypeCode_Case ('Exception_Head');
;    0139  2    TypeCode_Case ('Exception');
;    0140  2    TypeCode_Case ('Err_Halt');
;    0141  2    TypeCode_Case ('Summary');
;    0142  2    TypeCode_Case ('Program_Start');
;    0143  2    TypeCode_Case ('Program_End');
;    0144  2    TypeCode_Case ('First_Pass');
;    0145  2    TypeCode_Case ('No_Tests');
;    0146  2    TypeCode_Case ('Abort_Test');
;    0147  2    TypeCode_Case ('Abort_Program');
;    0148  2    TypeCode_Case ('Command_Err');
;    0149  2    TypeCode_Case ('Command_Out');
;    0150  2    TypeCode_Case ('Program_Info');
;    0151  2    TypeCode_Case ('Start_Err');
;    0152  2    TypeCode_Case ('Sequence_Error');
;    0153  2    TypeCode_Case ('CRD_AutoTest');
;    0154  2    TypeCode_Case ('Errprep');
;    0155  2    TypeCode_Case ('Param_Error');
;    0156  2    TypeCode_Case ('DS_Start');
;    0157  2    TypeCode_Case ('Script_Pnf');
;    0158  2    TypeCode_Case ('Script_Skip');
;    0159  2    TypeCode_Case ('Script_Prompt');
;    0160  2    TypeCode_Case ('Script_Echo');
;    0161  2    TypeCode_Case ('Qio_Nodriver');
;    0162  2    TypeCode_Case ('Qio_Wrongver');
;    0163  2    TypeCode_Case ('Qio_Invadp');
;    0164  2    TypeCode_Case ('Start_List');
;    0165  2
;    0166  2    [OUTRANGE]:
;    0167  2     $CRD_Print ($ASCIC ('Undefined typecode = !UL!/'), .TypeCode);
```

```
;   0168  2   TES;
;   0169  1 END;     ! Routine CRD$Print_TypeCode_Name


          .TITLE  CRDTYPCOD *** CRDTYPCOD Module
          .IDENT  \01-01\

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      44  4F  43  50  59  54  44  52  43  09  00000 P.AAA:  .ASCII  <9>\CRDTYPCOD\          ;
      2F  21  6C  61  72  65  6E  65  47  09  0000A P.AAB:  .ASCII  <9>\General!/\          ;
    2F  21  74  70  6D  6F  72  50  5F  53  44  0B  00014 P.AAC:  .ASCII  <11>\DS_Prompt!/\        ;
    2F  21  74  70  6D  6F  72  50  5F  72  65  73  55  0D  00020 P.AAD:  .ASCII  <13>\User_Prompt!/\       ;
21  72  6F  72  72  45  5F  6C  61  72  65  6E  65  47  0F  0002E P.AAE:  .ASCII  <15>\General_Error!/\        ;
        2F  0003D                ;
   2F  21  70  75  73  72  72  45  08  0003E P.AAF:  .ASCII  <8>\Errsup!/\           ;
   2F  21  73  79  73  72  72  45  08  00047 P.AAG:  .ASCII  <8>\Errsys!/\           ;
      2F  21  64  72  61  68  72  72  45  09  00050 P.AAH:  .ASCII  <9>\Errhard!/\          ;
      2F  21  74  66  6F  73  72  72  45  09  0005A P.AAI:  .ASCII  <9>\Errsoft!/\          ;
   2F  21  76  65  64  72  72  45  08  00064 P.AAJ:  .ASCII  <8>\Errdev!/\           ;
 2F  21  79  64  6F  42  5F  72  6F  72  72  45  0C  0006D P.AAK:  .ASCII  <12>\Error_Body!/\        ;
    2F  21  64  6E  45  5F  72  6F  72  72  45  0B  0007A P.AAL:  .ASCII  <11>\Error_End!/\         ;
64  61  65  48  5F  6E  6F  69  74  70  65  63  78  45  10  00086 P.AAM:  .ASCII  <16>\Exception_Head!/\       ;
          2F  21  00095            ;
    2F  21  6E  6F  69  74  70  65  63  78  45  0B  00097 P.AAN:  .ASCII  <11>\Exception!/\        ;
  2F  21  74  6C  61  48  5F  72  72  45  0A  000A3 P.AAO:  .ASCII  <10>\Err_Halt!/\         ;
      2F  21  79  72  61  6D  6D  75  53  09  000AE P.AAP:  .ASCII  <9>\Summary!/\          ;
21  74  72  61  74  53  5F  6D  61  72  67  6F  72  50  0F  000B8 P.AAQ:  .ASCII  <15>\Program_Start!/\        ;
        2F  000C7                ;
    2F  21  64  6E  45  5F  6D  61  72  67  6F  72  50  0D  000C8 P.AAR:  .ASCII  <13>\Program_End!/\         ;
 2F  21  73  73  61  50  5F  74  73  72  69  46  0C  000D6 P.AAS:  .ASCII  <12>\First_Pass!/\        ;
  2F  21  73  74  73  65  54  5F  6F  4E  0A  000E3 P.AAT:  .ASCII  <10>\No_Tests!/\         ;
 2F  21  74  73  65  54  5F  74  72  6F  62  41  0C  000EE P.AAU:  .ASCII  <12>\Abort_Test!/\        ;
21  6D  61  72  67  6F  72  50  5F  74  72  6F  62  41  0F  000FB P.AAV:  .ASCII  <15>\Abort_Program!/\        ;
        2F  0010A                ;
    2F  21  72  72  45  5F  64  6E  61  6D  6D  6F  43  0D  0010B P.AAW:  .ASCII  <13>\Command_Err!/\         ;
    2F  21  74  75  4F  5F  64  6E  61  6D  6D  6F  43  0D  00119 P.AAX:  .ASCII  <13>\Command_Out!/\         ;
2F  21  6F  66  6E  49  5F  6D  61  72  67  6F  72  50  0E  00127 P.AAY:  .ASCII  <14>\Program_Info!/\        ;
    2F  21  72  72  45  5F  74  72  61  74  53  0B  00136 P.AAZ:  .ASCII  <11>\Start_Err!/\        ;
72  6F  72  72  45  5F  65  63  6E  65  75  71  65  53  10  00142 P.ABA:  .ASCII  <16>\Sequence_Error!/\        ;
          2F  21  00151            ;
2F  21  74  73  65  54  6F  74  75  41  5F  44  52  43  0E  00153 P.ABB:  .ASCII  <14>\CRD_AutoTest!/\         ;
      2F  21  70  65  72  70  72  72  45  09  00162 P.ABC:  .ASCII  <9>\Errprep!/\          ;
    2F  21  72  6F  72  72  45  5F  6D  61  72  61  50  0D  0016C P.ABD:  .ASCII  <13>\Param_Error!/\         ;
  2F  21  74  72  61  74  53  5F  53  44  0A  0017A P.ABE:  .ASCII  <10>\DS_Start!/\         ;
 2F  21  66  6E  50  5F  74  70  69  72  63  53  0C  00185 P.ABF:  .ASCII  <12>\Script_Pnf!/\        ;
    2F  21  70  69  6B  53  5F  74  70  69  72  63  53  0D  00192 P.ABG:  .ASCII  <13>\Script_Skip!/\         ;
21  74  70  6D  6F  72  50  5F  74  70  69  72  63  53  0F  001A0 P.ABH:  .ASCII  <15>\Script_Prompt!/\         ;
        2F  001AF                ;
    2F  21  6F  68  63  45  5F  74  70  69  72  63  53  0D  001B0 P.ABI:  .ASCII  <13>\Script_Echo!/\         ;
2F  21  72  65  76  69  72  64  6F  4E  5F  6F  69  51  0E  001BE P.ABJ:  .ASCII  <14>\Qio_Nodriver!/\         ;
2F  21  72  65  76  67  6E  6F  72  57  5F  6F  69  51  0E  001CD P.ABK:  .ASCII  <14>\Qio_Wrongver!/\         ;
 2F  21  70  64  61  76  6E  49  5F  6F  69  51  0C  001DC P.ABL:  .ASCII  <12>\Qio_Invadp!/\        ;
 2F  21  74  73  69  4C  5F  74  72  61  74  53  0C  001E9 P.ABM:  .ASCII  <12>\Start_List!/\        ;
65  70  79  74  20  64  65  6E  69  66  65  64  6E  55  1A  001F6 P.ABN:  .ASCII  <26>\Undefined typecode = !UL!/\      ;
```

```
     2F  21  4C  55  21  20  3D  20  65  64  6F  63  00205            ;

          $MODULE=       P.AAA
           .EXTRN  CRD$PRINT_TYPECODE_NAME
           .EXTRN  CRD$PRINT

           .PSECT  CODE,NOWRT,  SHR,  PIC,2

             0004 00000     .ENTRY  CRD$PRINT_TYPECODE_NAME, Save R2       ; 0091
          52 00000000G  EF  D0 00002    MOVL    CRD$PRINT, R2            ; 0167
          25         00 04   AC  CF 00009    CASEL   TYPECODE, #0, #37         ; 0125
 0078        006F       0066     005D     0000E 1$:    .WORD   2$-1$,-          ;
 009C        0093       008A     0081     00016    3$-1$,-           ;
 00C0        00B7       00AE     00A5     0001E    4$-1$,-           ;
 00E4        00DB       00D2     00C9     00026    5$-1$,-           ;
 0108        00FF       00F6     00ED     0002E    6$-1$,-           ;
 012A        0122       011A     0111     00036    7$-1$,-           ;
 014A        0142       013A     0132     0003E    8$-1$,-           ;
 016A        0162       015A     0152     00046    9$-1$,-           ;
 018A        0182       017A     0172     0004E    10$-1$,-           ;
         019A    0192     00056    11$-1$,-          ;
            12$-1$,-          ;
            13$ 1$,-          ;
            14$-1$,-          ;
            15$ 1$,-          ;
            16$ 1$,           ;
            17$ 1$,           ;
            18$ 1$,           ;
            19$ 1$,           ;
            20$ 1$,           ;
            21$ 1$,           ;
            22$ 1$,           ;
            23$ 1$,           ;
            24$ 1$,           ;
            25$ 1$,           ;
            26$ 1$,           ;
            27$ 1$,           ;
            28$ 1$,           ;
            29$ 1$,           ;
            30$ 1$,           ;
            31$ 1$,           ;
            32$ 1$,           ;
            33$ 1$,           ;
            34$ 1$,           ;
            35$ 1$,           ;
            36$ 1$,           ;
            37$ 1$,           ;
            38$ 1$,           ;
            39$ 1$          ;
 04    AC  DD 0005A     PUSHL   TYPECODE          ; 0167
  00000000'  EF  9F 0005D    PUSHAB  P.ABN            ;
      01  DD 00063    PUSHL   #1           ;
      1A  DD 00065    PUSHL   #26           ;
      62       04  FB 00067    CALLS   #4, (R2)          ;
   04 0006A    RET                 ;
```

```
00000000'  EF  9F 0006B 2$:   PUSHAB  P.AAB          ; 0127
  013A  31 00071    BRW     40$                     ;
00000000'  EF  9F 00074 3$:   PUSHAB  P.AAC          ; 0128
  0131  31 0007A    BRW     40$                     ;
00000000'  EF  9F 0007D 4$:   PUSHAB  P.AAD          ; 0129
  0128  31 00083    BRW     40$                     ;
00000000'  EF  9F 00086 5$:   PUSHAB  P.AAE          ; 0130
  011F  31 0008C    BRW     40$                     ;
00000000'  EF  9F 0008F 6$:   PUSHAB  P.AAF          ; 0131
  0116  31 00095    BRW     40$                     ;
00000000'  EF  9F 00098 7$:   PUSHAB  P.AAG          ; 0132
  010D  31 0009E    BRW     40$                     ;
00000000'  EF  9F 000A1 8$:   PUSHAB  P.AAH          ; 0133
  0104  31 000A7    BRW     40$                     ;
00000000'  EF  9F 000AA 9$:   PUSHAB  P.AAI          ; 0134
  00FB  31 000B0    BRW     40$                     ;
00000000'  EF  9F 000B3 10$:   PUSHAB  P.AAJ          ; 0135
  00F2  31 000B9    BRW     40$                     ;
00000000'  EF  9F 000BC 11$:   PUSHAB  P.AAK          ; 0136
  00E9  31 000C2    BRW     40$                     ;
00000000'  EF  9F 000C5 12$:   PUSHAB  P.AAL          ; 0137
  00E0  31 000CB    BRW     40$                     ;
00000000'  EF  9F 000CE 13$:   PUSHAB  P.AAM          ; 0138
  00D7  31 000D4    BRW     40$                     ;
00000000'  EF  9F 000D7 14$:   PUSHAB  P.AAN          ; 0139
  00CE  31 000DD    BRW     40$                     ;
00000000'  EF  9F 000E0 15$:   PUSHAB  P.AAO          ; 0140
  00C5  31 000E6    BRW     40$                     ;
00000000'  EF  9F 000E9 16$:   PUSHAB  P.AAP          ; 0141
  00BC  31 000EF    BRW     40$                     ;
00000000'  EF  9F 000F2 17$:   PUSHAB  P.AAQ          ; 0142
  00B3  31 000F8    BRW     40$                     ;
00000000'  EF  9F 000FB 18$:   PUSHAB  P.AAR          ; 0143
  00AA  31 00101    BRW     40$                     ;
00000000'  EF  9F 00104 19$:   PUSHAB  P.AAS          ; 0144
  00A1  31 0010A    BRW     40$                     ;
00000000'  EF  9F 0010D 20$:   PUSHAB  P.AAT          ; 0145
  0098  31 00113    BRW     40$                     ;
00000000'  EF  9F 00116 21$:   PUSHAB  P.AAU          ; 0146
  008F  31 0011C    BRW     40$                     ;
00000000'  EF  9F 0011F 22$:   PUSHAB  P.AAV          ; 0147
  0086  31 00125    BRW     40$                     ;
00000000'  EF  9F 00128 23$:   PUSHAB  P.AAW          ; 0148
    7E  11 0012E    BRB     40$                     ;
00000000'  EF  9F 00130 24$:   PUSHAB  P.AAX          ; 0149
    76  11 00136    BRB     40$                     ;
00000000'  EF  9F 00138 25$:   PUSHAB  P.AAY          ; 0150
    6E  11 0013E    BRB     40$                     ;
00000000'  EF  9F 00140 26$:   PUSHAB  P.AAZ          ; 0151
    66  11 00146    BRB     40$                     ;
00000000'  EF  9F 00148 27$:   PUSHAB  P.ABA          ; 0152
    5E  11 0014E    BRB     40$                     ;
00000000'  EF  9F 00150 28$:   PUSHAB  P.ABB          ; 0153
    56  11 00156    BRB     40$                     ;
00000000'  EF  9F 00158 29$:   PUSHAB  P.ABC          ; 0154
```

```
        4E  11 0015E     BRB     40$             ;
00000000'  EF  9F 00160 30$:    PUSHAB  P.ABD           ; 0155
        46  11 00166     BRB     40$             ;
00000000'  EF  9F 00168 31$:    PUSHAB  P.ABE           ; 0156
        3E  11 0016E     BRB     40$             ;
00000000'  EF  9F 00170 32$:    PUSHAB  P.ABF           ; 0157
        36  11 00176     BRB     40$             ;
00000000'  EF  9F 00178 33$:    PUSHAB  P.ABG           ; 0158
        2E  11 0017E     BRB     40$             ;
00000000'  EF  9F 00180 34$:    PUSHAB  P.ABH           ; 0159
        26  11 00186     BRB     40$             ;
00000000'  EF  9F 00188 35$:    PUSHAB  P.ABI           ; 0160
        1E  11 0018E     BRB     40$             ;
00000000'  EF  9F 00190 36$:    PUSHAB  P.ABJ           ; 0161
        16  11 00196     BRB     40$             ;
00000000'  EF  9F 00198 37$:    PUSHAB  P.ABK           ; 0162
        0E  11 0019E     BRB     40$             ;
00000000'  EF  9F 001A0 38$:    PUSHAB  P.ABL           ; 0163
        06  11 001A6     BRB     40$             ;
00000000'  EF  9F 001A8 39$:    PUSHAB  P.ABM           ; 0164
        01  DD 001AE 40$:    PUSHL   #1              ;
        1A  DD 001B0     PUSHL   #26             ;
        62      03  FB 001B2    CALLS   #3, (R2)        ;
     04 001B5    RET                 ; 0169

; Routine Size:  438 bytes,    Routine Base:  CODE + 0000
```

```
;   0170  1 END     ! End of CRDTYPCOD module
;   0171  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          529  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          438  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
Symbol                    Type  Defined  Referenced ...
---- ------------------ -------- ----- ---------------
$ASCIC    Unbound           119
    Macro    Lib02   127     128     129     130     131     132     133     134     135     136     137
              138     139     140     141     142     143     144     145     146     147     148
              149     150     151     152     153     154     155     156     157     158     159
              160     161     162     163     164     167
$CRD_PRINT   Unbound           119
    Macro    Lib03   127     128     129     130     131     132     133     134     135     136     137
              138     139     140     141     142     143     144     145     146     147     148
              149     150     151     152     153     154     155     156     157     158     159
              160     161     162     163     164     167
$DS_TYPEDEF    Macro    Lib02     72     127     128     129     130     131     132     133     134     135     136
              137     138     139     140     141     142     143     144     145     146     147
              148     149     150     151     152     153     154     155     156     157     158
              159     160     161     162     163     164     167
$EQULST     Macro    Lib04    72     127     128     129     130     131     132     133     134     135     136
              137     138     139     140     141     142     143     144     145     146     147
              148     149     150     151     152     153     154     155     156     157     158
              159     160     161     162     163     164     167
$ER     Comptime     88
$MODULE     Bind          88
CRD$K_NUMB_TYPECODES  Literal  Lib03    123     125
CRD$PRINT    External Lib03   127ac   128ac   129ac   130ac   131ac   132ac   133ac   134ac   135ac   136ac   137ac
              138ac   139ac   140ac   141ac   142ac   143ac   144ac   145ac   146ac   147ac   148ac
              149ac   150ac   151ac   152ac   153ac   154ac   155ac   156ac   157ac   158ac   159ac
              160ac   161ac   162ac   163ac   164ac   167ac
CRD$PRINT_TYPECODE_NAME  GlobRout     91 Lib03e      68f
CRD_ASSERT    Macro    Lib03   122
DS$K_PRINTB    Literal      72
    Literal     127
    Literal     128
    Literal     129
    Literal     130
    Literal     131
    Literal     132
```

Symbol	Type	Defined	Referenced ...
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_PRINTF	Literal	72	
	Literal	127	127
	Literal	128	128
	Literal	129	129
	Literal	130	130
	Literal	131	131
	Literal	132	132
	Literal	133	133
	Literal	134	134
	Literal	135	135
	Literal	136	136
	Literal	137	137
	Literal	138	138
	Literal	139	139
	Literal	140	140
	Literal	141	141
	Literal	142	142
	Literal	143	143
	Literal	144	144
	Literal	145	145

Symbol	Type	Defined	Referenced ...
	Literal	146	146
	Literal	147	147
	Literal	148	148
	Literal	149	149
	Literal	150	150
	Literal	151	151
	Literal	152	152
	Literal	153	153
	Literal	154	154
	Literal	155	155
	Literal	156	156
	Literal	157	157
	Literal	158	158
	Literal	159	159
	Literal	160	160
	Literal	161	161
	Literal	162	162
	Literal	163	163
	Literal	164	164
	Literal	167	167
DS$K_PRINTI	Literal	72	
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	

Symbol	Type	Defined	Referenced ...
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_PRINTX	Literal	72	
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_ABORT_PROGRAM	Literal	72	147
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	

Symbol	Type	Defined	Referenced ...
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_ABORT_TEST	Literal	72	146
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	

Symbol	Type	Defined	Referenced ...
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_COMMAND_ERR	Literal	72	148
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	

Symbol	Type	Defined	Referenced ...
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_COMMAND_OUT	Literal	72	149
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_CRD_AUTOTEST	Literal	72	153
	Literal	127	127
	Literal	128	128
	Literal	129	129
	Literal	130	130

Symbol	Type	Defined	Referenced ...
Literal	131	131	
Literal	132	132	
Literal	133	133	
Literal	134	134	
Literal	135	135	
Literal	136	136	
Literal	137	137	
Literal	138	138	
Literal	139	139	
Literal	140	140	
Literal	141	141	
Literal	142	142	
Literal	143	143	
Literal	144	144	
Literal	145	145	
Literal	146	146	
Literal	147	147	
Literal	148	148	
Literal	149	149	
Literal	150	150	
Literal	151	151	
Literal	152	152	
Literal	153	153	
Literal	154	154	
Literal	155	155	
Literal	156	156	
Literal	157	157	
Literal	158	158	
Literal	159	159	
Literal	160	160	
Literal	161	161	
Literal	162	162	
Literal	163	163	
Literal	164	164	
Literal	167	167	
DS$K_TYPE_DS_PROMPT	Literal	72	128
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		

Symbol	Type	Defined	Referenced ...
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_DS_START	Literal	72	156
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	

```
Symbol                       Type  Defined  Referenced ...
---------------------------- ------- ----- ---------------
    Literal     157
    Literal     158
    Literal     159
    Literal     160
    Literal     161
    Literal     162
    Literal     163
    Literal     164
    Literal     167
DS$K_TYPE_ERRDEV  Literal      72   135
    Literal     127
    Literal     128
    Literal     129
    Literal     130
    Literal     131
    Literal     132
    Literal     133
    Literal     134
    Literal     135
    Literal     136
    Literal     137
    Literal     138
    Literal     139
    Literal     140
    Literal     141
    Literal     142
    Literal     143
    Literal     144
    Literal     145
    Literal     146
    Literal     147
    Literal     148
    Literal     149
    Literal     150
    Literal     151
    Literal     152
    Literal     153
    Literal     154
    Literal     155
    Literal     156
    Literal     157
    Literal     158
    Literal     159
    Literal     160
    Literal     161
    Literal     162
    Literal     163
    Literal     164
    Literal     167
DS$K_TYPE_ERRHARD  Literal      72   133
    Literal     127
    Literal     128
    Literal     129
```

Symbol	Type	Defined	Referenced ...
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		
DS$K_TYPE_ERROR_BODY	Literal	72	136
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		

Symbol	Type	Defined	Referenced ...
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_ERROR_END	Literal	72	137
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	

Symbol	Type	Defined	Referenced ...
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_ERRPREP	Literal	72	154
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_ERRSOFT	Literal	72	134
	Literal	127	
	Literal	128	

Symbol	Type	Defined	Referenced ...
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		
DS$K_TYPE_ERRSUP Literal	72	131	
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		

Symbol	Type	Defined	Referenced ...
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		
DS$K_TYPE_ERRSYS	Literal	72	132
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		

```
Symbol                      Type  Defined  Referenced ...
--------------------------- ------- ----- ---------------
    Literal     155
    Literal     156
    Literal     157
    Literal     158
    Literal     159
    Literal     160
    Literal     161
    Literal     162
    Literal     163
    Literal     164
    Literal     167
DS$K_TYPE_ERR_HALT  Literal      72   140
    Literal     127
    Literal     128
    Literal     129
    Literal     130
    Literal     131
    Literal     132
    Literal     133
    Literal     134
    Literal     135
    Literal     136
    Literal     137
    Literal     138
    Literal     139
    Literal     140
    Literal     141
    Literal     142
    Literal     143
    Literal     144
    Literal     145
    Literal     146
    Literal     147
    Literal     148
    Literal     149
    Literal     150
    Literal     151
    Literal     152
    Literal     153
    Literal     154
    Literal     155
    Literal     156
    Literal     157
    Literal     158
    Literal     159
    Literal     160
    Literal     161
    Literal     162
    Literal     163
    Literal     164
    Literal     167
DS$K_TYPE_EXCEPTION  Literal      72   139
    Literal     127
```

Symbol	Type	Defined	Referenced ...
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_EXCEPTION_HEAD	Literal	72	138
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	

Symbol	Type	Defined	Referenced ...
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_FIRST_PASS	Literal	72	144
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	

Symbol	Type	Defined	Referenced ...
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_GENERAL	Literal	72	127
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_GENERAL_ERROR	Literal	72	130

Symbol	Type	Defined	Referenced ...
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		
DS$K_TYPE_NO_TESTS	Literal	72	145
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		

Symbol	Type	Defined	Referenced ...
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		
DS$K_TYPE_PARAM_ERROR	Literal	72	155
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		

Symbol	Type	Defined	Referenced ...
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		
DS$K_TYPE_PROGRAM_END	Literal	72	143
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		

Symbol	Type	Defined	Referenced ...
DS$K_TYPE_PROGRAM_INFO	Literal	72	150
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_PROGRAM_START	Literal	72	142
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     139
    Literal     140
    Literal     141
    Literal     142
    Literal     143
    Literal     144
    Literal     145
    Literal     146
    Literal     147
    Literal     148
    Literal     149
    Literal     150
    Literal     151
    Literal     152
    Literal     153
    Literal     154
    Literal     155
    Literal     156
    Literal     157
    Literal     158
    Literal     159
    Literal     160
    Literal     161
    Literal     162
    Literal     163
    Literal     164
    Literal     167
DS$K_TYPE_QIO_INVADP  Literal      72   163
    Literal     127
    Literal     128
    Literal     129
    Literal     130
    Literal     131
    Literal     132
    Literal     133
    Literal     134
    Literal     135
    Literal     136
    Literal     137
    Literal     138
    Literal     139
    Literal     140
    Literal     141
    Literal     142
    Literal     143
    Literal     144
    Literal     145
    Literal     146
    Literal     147
    Literal     148
    Literal     149
    Literal     150
    Literal     151
```

Symbol	Type	Defined	Referenced ...
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_QIO_NODRIVER	Literal	72	161
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     167
DS$K_TYPE_QIO_WRONGVER  Literal      72   162
    Literal     127
    Literal     128
    Literal     129
    Literal     130
    Literal     131
    Literal     132
    Literal     133
    Literal     134
    Literal     135
    Literal     136
    Literal     137
    Literal     138
    Literal     139
    Literal     140
    Literal     141
    Literal     142
    Literal     143
    Literal     144
    Literal     145
    Literal     146
    Literal     147
    Literal     148
    Literal     149
    Literal     150
    Literal     151
    Literal     152
    Literal     153
    Literal     154
    Literal     155
    Literal     156
    Literal     157
    Literal     158
    Literal     159
    Literal     160
    Literal     161
    Literal     162
    Literal     163
    Literal     164
    Literal     167
DS$K_TYPE_SCRIPT_ECHO  Literal      72   160
    Literal     127
    Literal     128
    Literal     129
    Literal     130
    Literal     131
    Literal     132
    Literal     133
    Literal     134
    Literal     135
    Literal     136
    Literal     137
```

Symbol	Type	Defined	Referenced ...
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		
DS$K_TYPE_SCRIPT_PNF	Literal	72	157
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		

Symbol	Type	Defined	Referenced ...
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		
DS$K_TYPE_SCRIPT_PROMPT	Literal	72	159
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		

```
Symbol                        Type  Defined  Referenced ...
----------------------------- ------- ----- ---------------
    Literal     164
    Literal     167
DS$K_TYPE_SCRIPT_SKIP  Literal      72   158
    Literal     127
    Literal     128
    Literal     129
    Literal     130
    Literal     131
    Literal     132
    Literal     133
    Literal     134
    Literal     135
    Literal     136
    Literal     137
    Literal     138
    Literal     139
    Literal     140
    Literal     141
    Literal     142
    Literal     143
    Literal     144
    Literal     145
    Literal     146
    Literal     147
    Literal     148
    Literal     149
    Literal     150
    Literal     151
    Literal     152
    Literal     153
    Literal     154
    Literal     155
    Literal     156
    Literal     157
    Literal     158
    Literal     159
    Literal     160
    Literal     161
    Literal     162
    Literal     163
    Literal     164
    Literal     167
DS$K_TYPE_SEQUENCE_ERROR Literal      72   152
    Literal     127
    Literal     128
    Literal     129
    Literal     130
    Literal     131
    Literal     132
    Literal     133
    Literal     134
    Literal     135
    Literal     136
```

Symbol	Type	Defined	Referenced ...
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_START_ERR	Literal	72	151
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	

Symbol	Type	Defined	Referenced ...
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		
Literal	163		
Literal	164		
Literal	167		
DS$K_TYPE_START_LIST	Literal	72	164
Literal	127		
Literal	128		
Literal	129		
Literal	130		
Literal	131		
Literal	132		
Literal	133		
Literal	134		
Literal	135		
Literal	136		
Literal	137		
Literal	138		
Literal	139		
Literal	140		
Literal	141		
Literal	142		
Literal	143		
Literal	144		
Literal	145		
Literal	146		
Literal	147		
Literal	148		
Literal	149		
Literal	150		
Literal	151		
Literal	152		
Literal	153		
Literal	154		
Literal	155		
Literal	156		
Literal	157		
Literal	158		
Literal	159		
Literal	160		
Literal	161		
Literal	162		

Symbol	Type	Defined	Referenced ...
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_SUMMARY	Literal	72	141
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	
	Literal	136	
	Literal	137	
	Literal	138	
	Literal	139	
	Literal	140	
	Literal	141	
	Literal	142	
	Literal	143	
	Literal	144	
	Literal	145	
	Literal	146	
	Literal	147	
	Literal	148	
	Literal	149	
	Literal	150	
	Literal	151	
	Literal	152	
	Literal	153	
	Literal	154	
	Literal	155	
	Literal	156	
	Literal	157	
	Literal	158	
	Literal	159	
	Literal	160	
	Literal	161	
	Literal	162	
	Literal	163	
	Literal	164	
	Literal	167	
DS$K_TYPE_USER_PROMPT	Literal	72	129
	Literal	127	
	Literal	128	
	Literal	129	
	Literal	130	
	Literal	131	
	Literal	132	
	Literal	133	
	Literal	134	
	Literal	135	

ZCV
M-DDSST0TM

Symbol Type Defined Referenced ...

```
    Literal     136
    Literal     137
    Literal     138
    Literal     139
    Literal     140
    Literal     141
    Literal     142
    Literal     143
    Literal     144
    Literal     145
    Literal     146
    Literal     147
    Literal     148
    Literal     149
    Literal     150
    Literal     151
    Literal     152
    Literal     153
    Literal     154
    Literal     155
    Literal     156
    Literal     157
    Literal     158
    Literal     159
    Literal     160
    Literal     161
    Literal     162
    Literal     163
    Literal     164
    Literal     167
GET1ST_     Macro    Lib04    72     127     128     129     130     131     132     133     134     135     136
               137     138     139     140     141     142     143     144     145     146     147
               148     149     150     151     152     153     154     155     156     157     158
               159     160     161     162     163     164     167
GET2ND_     Unbound             72     127     128     129     130     131     132     133     134     135     136
               137     138     139     140     141     142     143     144     145     146     147
               148     149     150     151     152     153     154     155     156     157     158
               159     160     161     162     163     164     167
MODNAM     Macro    Lib01    88
NUL2ND_     Macro    Lib04    72     127     128     129     130     131     132     133     134     135     136
               137     138     139     140     141     142     143     144     145     146     147
               148     149     150     151     152     153     154     155     156     157     158
               159     160     161     162     163     164     167
TYPECODE    RoutForm     91    125.    167.
TYPECODE_CASE    Macro       117    127     128     129     130     131     132     133     134     135     136     137
               138     139     140     141     142     143     144     145     146     147     148
               149     150     151     152     153     154     155     156     157     158     159
               160     161     162     163     164
TYPECODE_NAME    MacrForm    117    118     119
```

```
  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]CRDTYPCOD.B32;1
    2  Module CRDTYPCOD
   55  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   58  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   61  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   64  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  171  Eludom CRDTYPCOD


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     @ Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;          -------- Symbols --------      Pages Processing
; File          Total    Loaded   Percent       Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0         46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923          2          0         94          00:00.1
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486          5          1         62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0       1000          00:04.4




;     COMMAND QUALIFIERS

; BLISS CRDTYPCOD.B32/LIST=CRDTYPCOD.LIS/CROSS_REFERENCE/DEBUG/OBJECT=CRDTYPCOD.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  438 code + 529 data bytes
; Run Time:    01:13.2
; Elapsed Time:    02:15.7
; Lines/CPU Min:     140
; Lexemes/CPU-Min: 83958
```

```
; Memory Used: 472 pages
; Compilation Complete
```

Table of contents

(1) 87 Libraries and Macros
(1) 93 The Dispatch Vectors

```
0000     1  .Title CRDVECS *** CRDVECS Module
0000     2  .Ident /V1.09/
0000     3  .NoShow Conditionals
0000     4
0000     5 ;++
0000     6 ; COPYRIGHT (C) 1977,1980,1984,1985
0000     7 ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
0000     8 ;
0000     9 ; THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
0000    10 ; COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
0000    11 ; ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
0000    12 ; MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
0000    13 ; EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
0000    14 ; TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
0000    15 ; REMAIN IN DEC.
0000    16 ;
0000    17 ; THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000    18 ; AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
0000    19 ; CORPORATION.
0000    20 ;
0000    21 ; DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
0000    22 ; SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
0000    23 ;--
```

```
                     ;++
                     ; Facility:
                     ;
                     ; VAX Diagnostic Supervisor (part of the CRD-Loadable image).
                     ;
                     ; Abstract:
                     ;
                     ; This module contains the dispatch vectors that the Resident-DS
                     ; routines need to access certain Loadable-CRD routines and locations (?).
                     ;
                     ; Environment:
                     ;
                     ; Not applicable
                     ;
                     ; Author:
                     ;
                     ; Jack Stansbury
                     ;
                     ; Date Written:
                     ;
                     ; March 30, 1982
                     ;
                     ; Modifications:
                     ;
                     ; 01 Jack Stansbury, June 21, 1982, Version 1.1
                     ;  Added CRD$Scan$Numeric and CRD$Exe$DeaNonPaged entry
                     ;  points for the Menu Test.
                     ;
                     ; 02 Jack Stansbury, July 27, 1982, Version 1.1
                     ;  Added the other (second) DS ^C handler address.
                     ;
                     ; 03 Jack Stansbury, August 19, 1982, Version 1.1
                     ;  Added two more dispatch vectors: Begin and CRD_Debug. Also
                     ;  changed the version number bytes from 1 to 2.
                     ;
                     ; 04 Jack Stansbury, September 10, 1982, Version 1.1
                     ;  Added the Boot$L_R5 address. This helps Menu Test decide
                     ;  from where it was invoked (by console, or by CRD CLI
                     ;  command).
                     ;
                     ; 05 Jack Stansbury, September 12, 1982, Version 1.1
                     ;  Took out the Boot$L_R5 longword (from above) and added the
                     ;  DSR$Check_MenuTest_Set and DSR$Check_AutoTest_Set routine
                     ;  addresses.
                     ;
                     ; 06 Jack Stansbury, March 3, 1983, Version ?.?
                     ;  Made changes similar to the changes in the DSVECS module to
                     ;  accomodate CRD's debug vectors.
                     ;
                     ; 07 John Ciukaj 5-Apr-1983 Version 1.2
                     ;  - Changed CRD$DSR$Check... vectors to distinguish between
                     ;    online and offline.
                     ;  - Changed revision level references [08] to [07]
                     ;
                     ; 08 Ron Parker 25-Feb-1985
```

```
0000     80 ;  - Added reference to DS$GA_PTDESC
0000     81 ;  - Added reference to DS$ATTACH
0000     82 ;
0000     83 ; 09 Amy Iorio 29-July-1985
0000     84 ;  Changed version comparison number from 3 to 10.
0000     85 ;--
```

```
                     0000     87  .SubTitle Libraries and Macros
                     0000     88
                     0000     89  .Library /Sys$Library:Lib/ ; Use Lib last
                     0000     90  .Library /$DS/    ; Use DS.Mar second
                     0000     91  .Library /$Diag/    ; Use Diag.Mar first
```

```
                        0000     93  .SubTitle The Dispatch Vectors
                    00000000     94  .Psect  $Base, Shr, NoExe, Wrt, Long
                        0000     95
                        0000     96  ;+
                        0000     97  ; These four bytes are for verification purposes. See the
                        0000     98  ; Exchange_Dispat_Vectors routine in the CRDLoad.B32 module for more
                        0000     99  ; of an explanation of these bytes. Note that these vectors are in the
                        0000    100  ; $Base Psect. This Psect will be the first one in the Loadable-CRD
                        0000    101  ; image, thus causing the values and addresses below to appear at the
                        0000    102  ; DS$GA_LastAdr address in the Supervisor when the CRD image file is
                        0000    103  ; read into the Supervisor. Note: the # of vectors in the first byte
                        0000    104  ; MUST agree with the value of the CRD$K_Total_Numb_Vectors constant
                        0000    105  ; in the CRD.R32 library module.
                        0000    106  ;-
                        0000    107
                        0000    108
                        0000    109 CRD$AL_CRD_Dispatch_Vectors::
                    19  0000    110  .Byte 25    ; The number of dispatch  [06]
                        0001    111       ; ... vectors   [06]
                    0A  0001    112  .Byte 10    ; The positive version number [06][09]
                    F6  0002    113  .Byte -10    ; The negative version number [06][09]
                    00  0003    114  .Byte 0    ; The null byte
                        0004    115
                        0004    116  ;+
                        0004    117  ; These are the actual dispatch vectors. The CRD-Loadable routines use
                        0004    118  ; the global label below to access the DS routines and DS locations.
                        0004    119  ; The CRD$DS_Interface routine is the routine that the DS Print routine
                        0004    120  ; will call when something is about to be printed. Since the Resident-DS
                        0004    121  ; does not need access to any other routines/locations in the
                        0004    122  ; Loadable-CRD, the other addresses are set to null. Note that there
                        0004    123  ; must be a fixed number of dispatch vectors, an equal number in both
                        0004    124  ; the Resident-DS (which requires 4), and in the Loadable-CRD.
                        0004    125  ;-
                        0004    126
                        0004    127 CRD$Exit_DS::     ; DSV$Exit routine
                00000000'  0004    128  .Address CRD$DS_Interface ; CRD's interface to the DS
                        0008    129
                        0008    130 CRD$Print::     ; DSX$Print routine
                00000000'  0008    131  .Address 0    ; No other addresses needed
                        000C    132
                        000C    133 CRD$GA_DS_Flags::     ; DS$GL_Flags longword
                00000000'  000C    134  .Address 0
                        0010    135
                        0010    136 CRD$GA_DS_Ctrl_C_First::    ; DS$GA_DS_Ctrl_C_First  [02]
                00000000'  0010    137  .Address 0    ; . . . longword  [02]
                        0014    138
                        0014    139 CRD$GA_PTable::     ; DS$GA_PTable longword
                00000000'  0014    140  .Address 0
                        0018    141
                        0018    142 CRD$Exe$AloNonPaged::     ; Exe$AloNonPaged rout'ne
                00000000'  0018    143  .Address 0
                        001C    144
                        001C    145 CRD$GA_User_Error_Text::    ; The address of the MsgAdr
                00000000'  001C    146  .Address 0    ; ... (from the ERRxxx macro) will
                        0020    147       ; ... be stored here.
```

```
                     0020   148
                     0020   149 CRD$Scan$Numeric::    ; Scan$Numeric routine   [01]
00000000'            0020   150  .Address 0   ; ... 'from the PARSE module) [01]
                     0024   151
                     0024   152 CRD$Exe$DeaNonPaged::    ; Exe$DeaNonPaged routine  [01]
00000000'            0024   153  .Address 0   ; ... (from the MEMMGT module) [01]
                     0028   154
                     0028   155 CRD$GA_DS_Ctrl_C_Second::   ; DS$GA_DS_Ctrl_C_Second [02]
00000000'            0028   156  .Address 0   ; . . . longword  [02]
                     002C   157
                     002C   158 CRD$GA_Begin::     ; The Begin label in the KERNEL [03]
00000000'            002C   159  .Address 0   ; . . . module.   [03]
                     0030   160
                     0030   161 CRD$GB_CRD_Debug::    ; For debugging CRD   [03]
00000034             0030   162  .BlkL  1   ; . . . Can set this from VDS [03]
                     0034   163       ; . . . This is byte-referenced [03]
                     0034   164
                     0034   165 CRD$DSR$Check_MenuTest_Off_Set::  ; The DSR$Check_MenuTest_Off_Set[07]
00000000'            0034   166  .Address 0   ; . . . routine from the KERNEL [05]
                     0038   167       ; . . . module.   [05]
                     0038   168
                     0038   169 CRD$DSR$Check_AutoTest_Off_Set::  ; The DSR$Check_AutoTest_Off_Set[07]
00000000'            0038   170  .Address 0   ; . . . routine from the KERNEL [05]
                     003C   171       ; . . . module.   [05]
                     003C   172
                     003C   173 CRD$DS$GA_PTDESC::    ; The DS$GA_PTDesc address [08]
00000000'            003C   174  .Address 0   ; . . . address from CRD_IMAGE  [08]
                     0040   175
                     0040   176  ;+
                     0040   177  ; These vectors are not currently used. They are here simply because  [07]
                     0040   178  ; the debug vectors require 25 such longwords to store all the debugging[07]
                     0040   179  ; information.       [07]
                     0040   180  ;-
                     0040   181
00000068             0040   182  .BlkL <25-15>    ; 25 = CRD$K_Total_Numb_Vectors [07]  .
                     0068   183       ; 15 = # of above vectors [07]
                     0068   184  .End
```

```
CRD$AL_CRD_DISPATCH_VECTORS      00000000 RG    01
CRD$DS$GA_PTDESC                 0000003C RG    01
CRD$DSR$CHECK_AUTOTEST_OFF_SET   00000038 RG    01
CRD$DSR$CHECK_MENUTEST_OFF_SET   00000034 RG    01
CRD$DS_INTERFACE                 ********   X   01
CRD$EXE$ALONONPAGED              00000018 RG    01
CRD$EXE$DEANONPAGED              00000024 RG    01
CRD$EXIT_DS                      00000004 RG    01
CRD$GA_BEGIN                     0000002C RG    01
CRD$GA_DS_CTRL_C_FIRST           00000010 RG    01
CRD$GA_DS_CTRL_C_SECOND          00000028 RG    01
CRD$GA_DS_FLAGS                  0000000C RG    01
CRD$GA_PTABLE                    00000014 RG    01
CRD$GA_USER_ERROR_TEXT           0000001C RG    01
CRD$GB_CRD_DEBUG                 00000030 RG    01
CRD$PRINT                        00000008 RG    01
CRD$SCAN$NUMERIC                 00000020 RG    01
```

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
$BASE	00000068 (104.)	01 (1.)	NOPIC USR CON REL LCL SHR NOEXE RD WRT NOVEC LONG

Symbol Cross Reference

SYMBOL	VALUE	DEFINITION	REFERENCES...
CRD$AL_CRD_DISPATCH_VECTORS	00000000-R	109 (1)	
CRDDSGA_PTDESC	0000003C-R	173 (1)	
CRDDSRCHECK_AUTOTEST_OFF_SET	00000038-R	169 (1)	
CRDDSRCHECK_MENUTEST_OFF_SET	00000034-R	165 (1)	
CRD$DS_INTERFACE	00000000-XR		128 (1)
CRDEXEALONONPAGED	00000018-R	142 (1)	
CRDEXEDEANONPAGED	00000024-R	152 (1)	
CRD$EXIT_DS	00000004-R	127 (1)	
CRD$GA_BEGIN	0000002C-R	158 (1)	
CRD$GA_DS_CTRL_C_FIRST	00000010-R	136 (1)	
CRD$GA_DS_CTRL_C_SECOND	00000028-R	155 (1)	
CRD$GA_DS_FLAGS	0000000C-R	133 (1)	
CRD$GA_PTABLE	00000014-R	139 (1)	
CRD$GA_USER_ERROR_TEXT	0000001C-R	145 (1)	
CRD$GB_CRD_DEBUG	00000030-R	161 (1)	
CRD$PRINT	00000008-R	130 (1)	
CRD$SCAN$NUMERIC	00000020-R	149 (1)	

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	110	00:00:00.19	00:00:01.27
Command processing	887	00:00:00.68	00:00:01.62
Pass 1	634	00:00:00.76	00:00:01.93
Symbol table sort	0	00:00:00.00	00:00:00.03
Pass 2	15	00:00:00.25	00:00:00.58
Symbol table output	2	00:00:00.02	00:00:00.02
Psect synopsis output	2	00:00:00.01	00:00:00.01
Cross-reference output	5	00:00:00.03	00:00:00.11
Assembler run totals	1657	00:00:01.95	00:00:05.59

The working set limit was 1862 pages.
1776 bytes (4 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 17 non-local and 0 local symbols.
184 source lines were read in Pass 1, producing 11 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.

```
                                    +---------------------------+
                                    ! Macro library statistics !
                                    +---------------------------+

Macro library name                                  Macros defined
------------------                                  --------------
DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.MLB;76                        0
DISK$DIAGPACK03:[DS.LIBRARIES]DS.MLB;37                          0
SYS$COMMON:[SYSLIB]LIB.MLB;1                                     0
SYS$COMMON:[SYSLIB]STARLET.MLB;1                                 0
TOTALS (all libraries)                                           0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO CRDVECS.MAR/LIS=CRDVECS.LIS/OBJ=CRDVECS.OBJ/CROSS
```

```
;    0001   0 %TITLE  'CRD MENU - Control C Handler'
;    0002   0 MODULE MENCNTLC  (
;    0003   0   IDENT = '01-00'
;    0004   0   ) =
;    0005   0 !
;    0006   0 ! COPYRIGHT (C)  1982
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 !
```

```
;    0024  0 !++
;    0025  0 ! FACILITY:
;    0026  0 !
;    0027  0 ! ABSTRACT:
;    0028  0 !
;    0029  0 ! ENVIRONMENT:
;    0030  0 !
;    0031  0 ! AUTHOR:
;    0032  0 !
;    0033  0 ! John Ciukaj  May-1982
;    0034  0 !
;    0035  0 ! CREATION DATE:
;    0036  0 !
;    0037  0 ! VERSION:
;    0038  0 !
;    0039  0 ! Modified by:
;    0040  0 !
;    0041  0 ! [01] Scott Cohen June 28, 1983 Version 1.2
;    0042  0 !  Added soft console support
;    0043  0 !--
;    0044  1 BEGIN      ! Module MENCNTLC
```

```
;    0045  1 %SBTTL 'Libraries'
;    0046  1
;    0047  1
;    0048  1  LIBRARY
;    0049  1    '$DS';
;    0050  1  LIBRARY
;    0051  1    '$DIAG';
;    0052  1  LIBRARY
;    0053  1    '$CRD';
;    0054  1  LIBRARY
;    0055  1    'SYS$LIBRARY:LIB';
```

```
;    0056  1 %SBTTL 'Forward-External  Routines'
;    0057  1
;    0058  1 FORWARD ROUTINE
;    0059  1     MEN$CntlC_Menu  : ADDRESSING_MODE (LONG_RELATIVE),
;    0060  1      ! The ^C handler routine for the CRD Menu Test
;    0061  1
;    0062  1     MEN$CntlCInit_Tbl Init : ADDRESSING_MODE (LONG_RELATIVE),
;    0063  1      ! Initialize some table
;    0064  1
;    0065  1     MEN$CntlC_Tbl_Init  : ADDRESSING_MODE (LONG_RELATIVE);
;    0066  1      ! Initialialize some other table
```

```
;    0067  1 %SBTTL 'Psect Definitions'
;    0068  1
;    0069  1 PSECT
;    0070  1     PLIT = Data (NOEXECUTE, SHARE,   NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0071  1
;    0072  1 PSECT
;    0073  1     CODE = Code (EXECUTE,   SHARE,   NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0074  1
;    0075  1 PSECT
;    0076  1     OWN  = Work (NOEXECUTE, NOSHARE, WRITE,   ADDRESSING_MODE (LONG_RELATIVE));
;    0077  1
;    0078  1 PSECT
;    0079  1     GLOBAL = Work (NOEXECUTE, NOSHARE, WRITE,   ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0080  1 %SBTTL 'Macro and Literal Definitions'
;    0081  1
;    0082  1  $DS_DSADef;    ! Define the DSA locations
;    0083  1  $CRD_Literals;    ! Define the CRD literals
;    0084  1  $MEN_Literals;    ! Define the Menu-specific literals
;    0085  1  $SCREEN_Literals;  ! Define the SCREEN soft console literals
```

```
;   0086  1 %SBTTL 'Module - Global Static Storage'
;   0087  1
;   0088  1      Modnam ('MENCNTLC');  ! Module Name for Errsup Macro
```

```
;    0089  1 %SBTTL 'Control C Initialization MENU Table'
;    0090  1
;    0091  1 !+
;    0092  1 ! Contains information relevant to all selections available
;    0093  1 ! from the Control C Initialization Menu Table.  This table is used
;    0094  1 ! for the Control C Menu selections for Control C's typed after
;    0095  1 ! CRD MENU is loaded and before the Functional Test Menu has been
;    0096  1 ! entered.
;    0097  1 ! Because the CRD MENU code is loadable it is necessary to invoke the
;    0098  1 ! Initialization routine associated with this table following loading
;    0099  1 ! and before the table is accessed (see MENTSTINI module).
;    0100  1 !-
```

```
;    0101  1 LITERAL
;    0102  1     MEN$GK_CntlC_InitTbl_Entries = 2;
;    0103  1
;    0104  1 OWN
;    0105  1     MEN$GA_CntlC_InitTbl: BLOCKVECTOR [ (MEN$GK_CntlC_InitTbl_Entries),
;    0106  1             MEN$K_CntlCTbl_Entry_Size, BYTE]
;    0107  1      FIELD ($MEN_CntlCTbl_Entry_FLD)
;    0108  1   INITIAL (
;    0109  1    BYTE (%C'A'),
;    0110  1    BYTE (MEN$K_CntlC_Sel_Exit),
;    0111  1    LONG (MEN$GT_CntlC_Sel_Exit),
;    0112  1
;    0113  1    BYTE (%C'B'),
;    0114  1    BYTE (MEN$K_CntlC_Sel_Continue),
;    0115  1    LONG (MEN$GT_CntlC_Sel_Continue)
;    0116  1
;    0117  1    );
```

```
;    0118  1 %SBTTL 'Control C Init. Menu Table Initialization Routine'
;    0119  1
;    0120  1 GLOBAL ROUTINE MEN$CntlCInit_Tbl_Init =
;    0121  1
;    0122  1 !++
;    0123  1 ! FUNCTIONAL DESCRIPTION:
;    0124  1 !
;    0125  1 !    Since the CRD MENU TEST control code is a loadable image,
;    0126  1 !    link-time resolved address pointers must have an offset added
;    0127  1 !    to them.  This is because the address pointers are resolved
;    0128  1 !    at link time with reference to a zero base address.  However,
;    0129  1 !    at run time the code will be loaded at an address starting at
;    0130  1 !    CRD$AL_CRD_Dispatch_Vectors.  Therefore, for all tables with text
;    0131  1 !    address pointers, add an offset equal to
;    0132  1 !    the value represented by the address location
;    0133  1 !    CRD$AL_CRD_Dispatch_Vectors!
;    0134  1 !
;    0135  1 ! INPUT PARAMETERS
;    0136  1 !
;    0137  1 ! OUTPUT PARAMETERS
;    0138  1 !
;    0139  1 ! IMPLICIT INPUTS
;    0140  1 !
;    0141  1 ! EXPLICIT OUTPUTS
;    0142  1 !
;    0143  1 ! SIDE EFFECTS
;    0144  1 !
;    0145  1 !    CntlC_InitTbl Init is specified True.
;    0146  1 !
;    0147  1 ! COMPLETION CODES
;    0148  1 !
;    0149  1 !    CRD$K_Success
;    0150  1 !--
```

```
;    0151  2 BEGIN     ! Routine MEN$CntlCInit_Tbl_Init
;    0152  2     REGISTER
;    0153  2   Text_Ptr : REF VECTOR [, LONG],
;    0154  2   Index;
;    0155  2
;    0156  2     OWN
;    0157  2         Cntlc_InitTbl_Init : BYTE INITIAL (BYTE (False));
;    0158  2       ! Flag location indicating whether
;    0159  2       ! Table Text Address pointers
;    0160  2       ! have been initialized with
;    0161  2       ! offset address  values to account
;    0162  2       ! for the fact that the code
;    0163  2       ! is loaded at a dynamic address
;    0164  2       ! location.
;    0165  2       ! 1 or True  = Table Initialized
;    0166  2       ! 0 or False = Table not initialized
;    0167  2
;    0168  2     !+
;    0169  2     ! Perform initialization only once
;    0170  2     !-
;    0171  2
;    0172  2     IF (NOT .Cntlc_InitTbl_Init) THEN
;    0173  3   BEGIN
;    0174  3   !+
;    0175  3   ! Initialize Address Pointers
;    0176  3   !-
;    0177  3
;    0178  3   Index = -1;
;    0179  3   WHILE ((Index = .Index + 1) LSS MEN$GK_Cntlc_InitTbl_Entries) DO
;    0180  4       BEGIN
;    0181  4       Text_Ptr = MEN$GA_CntlC_InitTbl [.Index, CntlcTbl$L_Entry_Text];
;    0182  4       ! Address of text pointer
;    0183  4       Text_Ptr [0] = .Text_Ptr [0] + CRD$AL_CRD_Dispatch_Vectors;
;    0184  4       ! Add offset
;    0185  3       END;
;    0186  3   Cntlc_InitTbl_Init = True; ! Indicate that Initialization is done
;    0187  2   END;
;    0188  2     RETURN (CRD$K_Success);
;    0189  1 END;     ! Routine MEN$CntlCInit_Tbl_Init


        .TITLE  MENCNTLC CRD MENU - Control C Handler
        .IDENT  \01-00\

        .PSECT  WORK,NOEXE,2

    41  00000 MEN$GA_CNTLC_INITTBL:
        .BYTE   65              ;
    01  00001     .BYTE   1               ;
     00000000G 00002     .ADDRESS MEN$GT_CNTLC_SEL_EXIT          ;
    42  00006     .BYTE   66              ;
    02  00007     .BYTE   2               ;
     00000000G 00008     .ADDRESS MEN$GT_CNTLC_SEL_CONTINUE          ;
    00  0000C CNTLC_INITTBL_INIT:
        .BYTE   0               ;
```

```
                .PSECT  DATA,NOWRT,NOEXE,  SHR,2

   43  4C  54  4E  43  4E  45  4D  08  00000 P.AAA:  .ASCII  <8>\MENCNTLC\          ;

            DSA$AL_APTMAIL=     65024
            DSA$GL_FLAGS=       65024
            DSA$GL_APTCOM=      65028
            DSA$GL_PASSES=      65032
            DSA$GL_UNITS=       65036
            DSA$GL_SECTNO=      65040
            DSA$GL_SID=        65044
            DSA$GL_MSGTYP=      65088
            DSA$GL_ERRNO=       65092
            DSA$GL_EVENT=       65096
            DSA$GL_SUBTNO=      65100
            DSA$GL_TESTNO=      65104
            DSA$GL_PASSNO=      65108
            DSA$GL_DEVLEN=      65112
            DSA$GL_DEVNAM=      65116
            DSA$GL_MSGPTR=      65128
            $MODULE=        P.AAA
             .EXTRN  MEN$CNTLC_MENU, MEN$CNTLCINIT_TBL_INIT
             .EXTRN  MEN$CNTLC_TBL_INIT
             .EXTRN  MEN$GT_CNTLC_SEL_EXIT
             .EXTRN  MEN$GT_CNTLC_SEL_CONTINUE
             .EXTRN  CRD$AL_CRD_DISPATCH_VECTORS

             .PSECT  CODE,NOWRT,  SHR,  PIC,2

               0004 00000     .ENTRY  MEN$CNTLCINIT_TBL_INIT, Save R2       ; 0120
            29 00000000'  EF  E8 00002     BLBS    CNTLC_INITTBL_INIT, 3$        ; 0172
            51       01  CE 00009     MNEGL   #1, INDEX             ; 0178
            16  11 0000C     BRB     2$                ; 0179
            52       51       06  C5 0000E 1$:   MULL3   #6, INDEX, R2          ; 0181
            50 00000000'EF42  9E 00012     MOVAB   MEN$GA_CNTLC_INITTBL+2[R2], TEXT_PTR       ;
            52 00000000G  EF  9E 0001A     MOVAB   CRD$AL_CRD_DISPATCH_VECTORS, R2       ; 0183
            60       52  C0 00021     ADDL2   R2, (TEXT_PTR)          ;
            51  D6 00024 2$:   INCL    INDEX             ; 0179
            02       51  D1 00026     CMPL    INDEX, #2             ;
            E3  19 00029     BLSS    1$                ;
         00000000'  EF       01  90 0002B     MOVB    #1, CNTLC_INITTBL_INIT        ; 0186
            50       01  D0 00032 3$:   MOVL    #1, R0              ; 0188
             04 00035     RET                ; 0189

; Routine Size:  54 bytes,    Routine Base:  CODE + 0000
```

```
;    0190  1 %SBTTL 'Control C MENU Table'
;    0191  1 !+
;    0192  1 ! Contains information relevant to all selections available
;    0193  1 ! from the Control C  Menu Table.  This table is used
;    0194  1 ! for the Control C Menu selections for Control C's typed after
;    0195  1 ! the Functional Test Menu has been
;    0196  1 ! entered.
;    0197  1 ! Because the CRD MENU code is loadable it is necessary to invoke the
;    0198  1 ! Initialization routine associated with this table following loading
;    0199  1 ! and before the table is accessed (see MENTSTINI module).
;    0200  1 !-
```

```
;    0201  1 LITERAL
;    0202  1     MEN$GK_CntlC_Tbl_Entries = 3;
;    0203  1
;    0204  1 OWN
;    0205  1     MEN$GA_CntlC_Tbl:   BLOCKVECTOR [ (MEN$GK_CntlC_Tbl_Entries),
;    0206  1             MEN$K_CntlCTbl_Entry_Size, BYTE]
;    0207  1    FIELD ($MEN_CntlCTbl_Entry_FLD)
;    0208  1   INITIAL (
;    0209  1    BYTE (%C'A'),
;    0210  1    BYTE (MEN$K_CntlC_Sel_Exit),
;    0211  1    LONG (MEN$GT_CntlC_Sel_Exit),
;    0212  1
;    0213  1    BYTE (%C'B'),
;    0214  1    BYTE (MEN$K_CntlC_Sel_FTMenu),
;    0215  1    LONG (MEN$GT_CntlC_Sel_FTMenu),
;    0216  1
;    0217  1    BYTE (%C'C'),
;    0218  1    BYTE (MEN$K_CntlC_Sel_Continue),
;    0219  1    LONG (MEN$GT_CntlC_Sel_Continue)
;    0220  1
;    0221  1    );
```

```
;    0222  1 %SBTTL 'Control C Menu Table Initialization Routine'
;    0223  1
;    0224  1 GLOBAL ROUTINE MEN$CntlC_Tbl_Init =
;    0225  1
;    0226  1 !++
;    0227  1 ! FUNCTIONAL DESCRIPTION:
;    0228  1 !
;    0229  1 !    Since the CRD MENU TEST control code is a loadable image,
;    0230  1 !    link-time resolved address pointers must have an offset added
;    0231  1 !    to them.  This is because the address pointers are resolved
;    0232  1 !    at link time with reference to a zero base address.  However,
;    0233  1 !    at run time the code will be loaded at an address starting at
;    0234  1 !    CRD$AL_CRD_Dispatch_Vectors.  Therefore, for all tables with text
;    0235  1 !    address pointers, add an offset equal to
;    0236  1 !    the value represented by the address location
;    0237  1 !    CRD$AL_CRD_Dispatch_Vectors!
;    0238  1 !
;    0239  1 ! INPUT PARAMETERS
;    0240  1 !
;    0241  1 ! OUTPUT PARAMETERS
;    0242  1 !
;    0243  1 ! IMPLICIT INPUTS
;    0244  1 !
;    0245  1 ! EXPLICIT OUTPUTS
;    0246  1 !
;    0247  1 ! SIDE EFFECTS
;    0248  1 !
;    0249  1 !    CntlC_Tbl_Init is specified True.
;    0250  1 !
;    0251  1 ! COMPLETION CODES
;    0252  1 !
;    0253  1 !    CRD$K_Success
;    0254  1 !--
```

```
;    0255  2 BEGIN      ! Routine MEN$CntlC_Tbl_Init
;    0256  2     LOCAL
;    0257  2  Text_Ptr : REF VECTOR [, LONG],
;    0258  2  Index;
;    0259  2
;    0260  2     OWN
;    0261  2         Cntlc_Tbl_Init : BYTE INITIAL (BYTE (False));
;    0262  2       ! Flag location indicating whether
;    0263  2       ! Table Text Address pointers
;    0264  2       ! have been initialized with
;    0265  2       ! offset address  values to account
;    0266  2       ! for the fact that the code
;    0267  2       ! is loaded at a dynamic address
;    0268  2       ! location.
;    0269  2       ! 1 or True  = Table Initialized
;    0270  2       ! 0 or False = Table not initialized
;    0271  2
;    0272  2     !+
;    0273  2     ! Perform initialization only once
;    0274  2     !-
;    0275  2
;    0276  3     IF (NOT .Cntlc_Tbl_In't)
;    0277  2     THEN
;    0278  3  BEGIN
;    0279  3  !+
;    0280  3  ! Initialize Address Pointers
;    0281  3  !-
;    0282  3
;    0283  3  Index = -1;
;    0284  3  WHILE ((Index = .Index + 1) LSS MEN$GK_Cntlc_Tbl_Entries) DO
;    0285  4      BEGIN
;    0286  4      Text_Ptr = MEN$GA_CntlC_Tbl [.Index, CntlcTbl$L_Entry_Text];
;    0287  4      ! Address of text pointer
;    0288  4      Text_Ptr [0] = .Text_Ptr [0] + CRD$AL_CRD_Dispatch_Vectors;
;    0289  4      ! Add offset
;    0290  3      END;
;    0291  3  Cntlc_Tbl_Init = True;  ! Indicate that Initialization is done
;    0292  2  END;
;    0293  2     RETURN (CRD$K_Success);
;    0294  1 END;     ! Routine MEN$CntlC_Tbl_Init


          .PSECT  WORK,NOEXE,2

          0000D    .BLKB   3
     41   00010 MEN$GA_CNTLC_TBL:
          .BYTE   65              ;
     01   00011     .BYTE   1              ;
      00000000G 00012     .ADDRESS MEN$GT_CNTLC_SEL_EXIT         ;
     42   00016     .BYTE   66             ;
     03   00017     .BYTE   3              ;
      00000000G 00018     .ADDRESS MEN$GT_CNTLC_SEL_FTMENU         ;
     43   0001C     .BYTE   67             ;
     02   0001D     .BYTE   2              ;
```

```
         00000000G 0001E     .ADDRESS MEN$GT_CNTLC_SEL_CONTINUE        ;
        00  00022 CNTLC_TBL_INIT:
            .BYTE   0           ;

            .EXTRN  MEN$GT_CNTLC_SEL_FTMENU

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0004 00000     .ENTRY  MEN$CNTLC_TBL_INIT, Save R2        ; 0224
            29 00000000'  EF  E8 00002     BLBS    CNTLC_TBL_INIT, 3$          ; 0276
            50       01  CE 00009     MNEGL   #1, INDEX           ; 0283
            16  11 0000C     BRB     2$              ; 0284
            51        50       06  C5 0000E 1$:    MULL3   #6, INDEX, R1           ; 0286
            52 00000000'EF41  9E 00012     MOVAB   MEN$GA_CNTLC_TBL+2[R1], TEXT_PTR        ;
            51 00000000G  EF  9E 0001A     MOVAB   CRD$AL_CRD_DISPATCH_VECTORS, R1         ; 0288
            62       51  C0 00021     ADDL2   R1, (TEXT_PTR)          ;
            50  D6 00024 2$:    INCL    INDEX           ; 0284
            03       50  D1 00026     CMPL    INDEX, #3           ;
            E3  19 00029     BLSS    1$              ;
        00000000'  EF       01  90 0002B     MOVB    #1, CNTLC_TBL_INIT          ; 0291
            50       01  D0 00032 3$:    MOVL    #1, R0          ; 0293
             04 00035     RET                 ; 0294

; Routine Size:  54 bytes,    Routine Base:  CODE + 0036
```

```
;    0295  1 %SBTTL 'Control C Menu Handler Routine'
;    0296  1
;    0297  1 GLOBAL ROUTINE MEN$CntlC_Menu =
;    0298  1
;    0299  1 !++
;    0300  1 ! FUNCTIONAL DESCRIPTION:
;    0301  1 !
;    0302  1 ! INPUT PARAMETERS
;    0303  1 !
;    0304  1 ! OUTPUT PARAMETERS
;    0305  1 !
;    0306  1 ! IMPLICIT INPUTS
;    0307  1 !
;    0308  1 ! EXPLICIT OUTPUTS
;    0309  1 !
;    0310  1 ! SIDE EFFECTS
;    0311  1 !
;    0312  1 !    If operator chooses 'Exit' from menu then CRD MENU TEST is aborted
;    0313  1 !    by calling CRD$Stop.
;    0314  1 !
;    0315  1 ! COMPLETION CODES
;    0316  1 !
;    0317  1 !    CRD$K_Success
;    0318  1 !
;    0319  1 !--
```

```
;    0320  2 BEGIN     ! Routine MEN$CNTLC_MENU
;    0321  2     REGISTER
;    0322  2  Entry_Ptr : REF $MEN_CntlCTbl_Entry_ATTR,
;    0323  2  CntlC_Tbl_Index,
;    0324  2  Prompt  : BYTE,
;    0325  2  Oper  : BYTE,
;    0326  2  Valid_Status;
;    0327  2
;    0328  2     LOCAL
;    0329  2  CntlC_Tbl_Entries,
;    0330  2  CntlC_Tbl_Ptr   : REF BLOCKVECTOR [, MEN$K_CntlCTbl_Entry_Size, BYTE]
;    0331  2        FIELD ($MEN_CntlCTbl_Entry_Fld),
;    0332  2  Valid_Alpha_Upper : VECTOR [CH$ALLOCATION (1)],
;    0333  2  Valid_Alpha_Lower : VECTOR [CH$ALLOCATION (1)];
;    0334  2
;    0335  2     LITERAL
;    0336  2  OperInput_Buffer_Sz = 132;
;    0337  2
;    0338  2     LOCAL
;    0339  2  OperInput_Buffer : VECTOR [OperInput_Buffer_Sz, BYTE];
;    0340  2
;    0341  2     BIND
;    0342  2  T_Prompt_Default = $ASCIC ('?') : VECTOR [,BYTE];
;    0343  2
;    0344  2     !+
;    0345  2     !-----------------------------------------------------------------
;    0346  2     ! LOCAL ROUTINE:
;    0347  2     ! Type_Control_C_Menu
;    0348  2     !  Type the options for the control-c menu
;    0349  2     !-----------------------------------------------------------------
;    0350  2     !-
;    0351  2
;    0352  2     ROUTINE Type_Control_C_Menu (Control_C_Table_Ptr, Control_C_Table_Entries) : NOVALUE =
;    0353  3     BEGIN
;    0354  3  REGISTER
;    0355  3   CntlC_Tbl_Index,
;    0356  3   Select_Alpha_Ptr,
;    0357  3   Select_Text_Ptr;
;    0358  3
;    0359  3  BIND
;    0360  3   CntlC_Tbl_Ptr   = (.Control_C_Table_Ptr),
;    0361  3   CntlC_Tbl_Entries = (.Control_C_Table_Entries);
;    0362  3
;    0363  3  MAP
;    0364  3   CntlC_Tbl_Ptr : REF BLOCKVECTOR [, MEN$K_CntlCTbl_Entry_Size, BYTE]
;    0365  3     FIELD ($MEN_CntlCTbl_Entry_Fld);
;    0366  3
;    0367  3  !+
;    0368  3  ! Print the Control-C Menu title
;    0369  3  !-
;    0370  3
;    0371  3  $CRD_Message (New_Line, MEN$GT_CntlC_Menu_Title);
;    0372  3
;    0373  3  !+
;    0374  3  ! Print the Menu Selections from the appropriate Control C table
```

```
;    0375  3  !-
;    0376  3
;    0377  4  IF (NOT .MEN$GB_FncTst_Init)
;    0378  3  THEN
;    0379  4   BEGIN
;    0380  4   CntlC_Tbl_Ptr    = MEN$GA_CntlC_InitTbl;
;    0381  4   CntlC_Tbl_Entries = MEN$GK_CntlC_InitTbl_Entries;
;    0382  4   END
;    0383  3  ELSE
;    0384  4   BEGIN
;    0385  4   CntlC_Tbl_Ptr     = MEN$GA_CntlC_Tbl;
;    0386  4   CntlC_Tbl_Entries = MEN$GK_CntlC_Tbl_Entries;
;    0387  3   END;
;    0388  3
;    0389  3  CntlC_Tbl_Index = -1;
;    0390  3
;    0391  3  WHILE ((CntlC_Tbl_Index = .CntlC_Tbl_Index + 1) LSS .CntlC_Tbl_Entries) DO
;    0392  4   BEGIN
;    0393  4   Select_Alpha_Ptr = CntlC_Tbl_Ptr [.CntlC_Tbl_Index, CntlCTbl$B_Entry_Alpha];
;    0394  4   Select_Text_Ptr  = .CntlC_Tbl_Ptr [.CntlC_Tbl_Index, CntlCTbl$L_Entry_Text];
;    0395  4
; P  0396  4   $CRD_Message (New_Line,
; P  0397  4          MEN$GT_CntlC_Menu_Selection,
; P  0398  4          1,   ! Length of selection choice Alpha
; P  0399  4          .Select_Alpha_Ptr, ! Address of Alpha
; P  0400  4          .Select_Text_Ptr  ! Address of selection text ASCIC
;    0401  4          );
;    0402  3   END;
;    0403  3
;    0404  3  $CRD_Message (New_Line, MEN$GT_CntlCM_Choice);
;    0405  2     END;    ! Routine Type_Control_C_Menu



        .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      3F  01  00009 P.AAB:  .ASCII  <1>\?\          ;

      T_PROMPT_DEFAULT=   P.AAB
        .EXTRN  MEN$GT_CNTLC_MENU_TITLE
        .EXTRN  CRD$GT_NEW_LINE_MESSAGE
        .EXTRN  CRD$PRINT, MEN$GB_FNCTST_INIT
        .EXTRN  MEN$GT_CNTLC_MENU_SELECTION
        .EXTRN  MEN$GT_CNTLCM_CHOICE

        .PSECT  CODE,NOWRT,  SHR,  PIC,2

          001C 00000 TYPE_CONTROL_C_MENU:
        .WORD   Save R2,R3,R4          ; 0352
  00000000G  EF  9F 00002    PUSHAB  CRD$GT_NEW_LINE_MESSAGE        ; 0371
      01  DD 00008    PUSHL   #1             ;
      1A  DD 0000A    PUSHL   #26            ;
  00000000G  FF       03  FB 0000C    CALLS   #3, @CRD$PRINT         ;
  00000000G  EF  9F 00013    PUSHAB  MEN$GT_CNTLC_MENU_TITLE        ;
      01  DD 00019    PUSHL   #1             ;
```

```
                1A  DD 0001B    PUSHL   #26             ;
         00000000G  FF      03  FB 0001D    CALLS   #3, aCRD$PRINT          ;
                0E 00000000G  EF  E8 00024    BLBS    MEN$GB_FNCTST_INIT, 1$          ; 0377
            04  BC 00000000'  EF  9E 0002B    MOVAB   MEN$GA_CNTLC_INITTBL, aCONTROL_C_TABLE_PTR  ; 0380
            08  BC      02  D0 00033    MOVL    #2, aCONTROL_C_TABLE_ENTRIES        ; 0381
                0C  11 00037    BRB     2$              ; 0377
            04  BC 00000000'  EF  9E 00039 1$:   MOVAB   MEN$GA_CNTLC_TBL, aCONTROL_C_TABLE_PTR      ; 0385
            08  BC      03  D0 00041    MOVL    #3, aCONTROL_C_TABLE_ENTRIES        ; 0386
                52      01  CE 00045 2$:   MNEGL   #1, CNTLC_TBL_INDEX          ; 0389
                39  11 00048    BRB     4$              ; 0391
                50 04   BC  D0 0004A 3$:   MOVL    aCONTROL_C_TABLE_PTR, R0        ; 0393
                51        52      06  C5 0004E    MULL3   #6, CNTLC_TBL_INDEX, R1         ;
                53        50      51  C1 00052    ADDL3   R1, R0, SELECT_ALPHA_PTR        ;
            02 A140  9F 00056    PUSHAB  2(R1)[R0]           ; 0394
                54      9E  D0 0005A    MOVL    @(SP)+, SELECT_TEXT_PTR         ;
         00000000G  EF  9F 0005D    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0401
                01  DD 00063    PUSHL   #1              ;
                1A  DD 00065    PUSHL   #26             ;
         00000000G  FF      03  FB 00067    CALLS   #3, aCRD$PRINT          ;
                18  BB 0   5E    PUSHR   #^M<R3,R4>          ;
                01  DD 0UU70    PUSHL   #1              ;
         00000000G  EF  9F 00072    PUSHAB  MEN$GT_CNTLC_MENU_SELECTION         ;
                01  DD 00078    PUSHL   #1              ;
                1A  DD 0007A    PUSHL   #26             ;
         00000000G  FF      06  FB 0007C    CALLS   #6, aCRD$PRINT          ;
                52  D6 00083 4$:   INCL    CNTLC_TBL_INDEX         ; 0391
            08  BC      52  D1 00085    CMPL    CNTLC_TBL_INDEX, aCONTROL_C_TABLE_ENTRIES   ;
                BF  19 00089    BLSS    3$              ;
         00000000G  EF  9F 0008B    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0404
                01  DD 00091    PUSHL   #1              ;
                1A  DD 00093    PUSHL   #26             ;
         00000000G  FF      03  FB 00095    CALLS   #3, aCRD$PRINT          ;
         00000000G  EF  9F 0009C    PUSHAB  MEN$GT_CNTLCM_CHOICE          ;
                01  DD 000A2    PUSHL   #1              ;
                1A  DD 000A4    PUSHL   #26             ;
         00000000G  FF      03  FB 000A6    CALLS   #3, aCRD$PRINT          ;
          04 000AD    RET                 ; 0405

; Routine Size:  174 bytes,    Routine Base:  CODE + 006C


;   0406  2
;   0407  2     !+
;   0408  2     ! ----------------------------------- ----------------------------------
;   0409  2     ! MAIN ROUTINE:
;   0410  2     ! MEN$Cntlc_Menu
;   0411  2     !-----------------------------------------------------------------------
;   0412  2     !-
;   0413  2
;   0414  2     CRD$Disable_Ctrl_C ();  ! Disable catching ^C's
;   0415  2     CRD$Clear_Ctrl_C_Handlers (); ! Clear out the handler routine addresses
;   0416  2     CRD$Clear_Ctrl_C_Flag ();  ! Clear the DS$GL_Flags <DS$V_CtrlC> flag
;   0417  2
;   0418  2     !+
;   0419  2     ! Prompt for operator input until a valid menu selection is entered
```

```
;    0420  2     !-
;    0421  2
;    0422  2     Valid_Status = False;
;    0423  2
;    0424  2     DO
;    0425  3   BEGIN
;    0426  3   Type_Control_C_Menu (CntlC_Tbl_Ptr, CntlC_Tbl_Entries);
;    0427  3
;    0428  3   Prompt = .DSA$V_Prompt;
;    0429  3   Oper   = .DSA$V_Oper;
;    0430  3
;    0431  3   CRD$GB_User_Prompt_Okay = True; ! Allow User_Prompt typecode to pass through CRD untouched.
;    0432  3   DSA$V_Prompt = On;  ! Turn on PROMPTing to allow prompts from the operator
;    0433  3   DSA$V_Oper   = On;  ! Turn on OPER to allow prompts from the operator
;    0434  3
; P  0435  4   IF ($MEN_ASKSTR (Msg_Adr = MEN$GT_CntlC_Menu_Prompt,
; P  0436  4      Buf_Adr = OperInput_Buffer,
;    0437  4      Def_Adr = T_Prompt_Default))
;    0438  3   THEN
;    0439  4       BEGIN
;    0440  4       CntlC_Tbl_Index = -1;
;    0441  4
;    0442  5       WHILE ((CntlC_Tbl_Index = .CntlC_Tbl_Index + 1) LSS (.CntlC_Tbl_Entries) AND
;    0443  4       (NOT .Valid_Status)) DO
;    0444  5    BEGIN
;    0445  5    Entry_Ptr = CntlC_Tbl_Ptr [.CntlC_Tbl_Index, CntlCTbl$B_Entry_Alpha];
;    0446  5    Valid_Alpha_Upper = .Entry_Ptr [CntlCTbl$B_Entry_Alpha];
;    0447  5    Valid_Alpha_Lower = .Valid_Alpha_Upper + %X'20';
;    0448  5
;    0449  5    Valid_Status = CH$EQL (1, Valid_Alpha_Upper, .OperInput_Buffer [0], OperInput_Buffer [1]) OR
;    0450  5           CH$EQL (1, Valid_Alpha_Lower, .OperInput_Buffer [0], OperInput_Buffer [1]);
;    0451  4    END;
;    0452  3       END;
;    0453  3
;    0454  4   IF (NOT .Valid_Status)
;    0455  3   THEN
;    0456  3       !+
;    0457  3       ! If <CRLF> or ^C was typed don't type message, just prompt again.
;    0458  3       ! Both of these operator inputs will load the operator buffer
;    0459  3       ! with a default.
;    0460  3       !-
;    0461  3
;    0462  4       IF (NOT CH$EQL (.OperInput_Buffer [0], OperInput_Buffer [1],
;    0463  4         .T_Prompt_Default [0], T_Prompt_Default [1] ))
;    0464  3       THEN
;    0465  3        $CRD_Message ( New_Line, MEN$GT_InvOperInput);
;    0466  3   END     ! DO ... BEGIN ...
;    0467  2      UNTIL (.Valid_Status);  ! Repeat till we get a good answer
;    0468  2
;    0469  2      CRD$GB_User_Prompt_Okay = False; ! Disallow User_Prompt typecode to pass through CRD untouched.
;    0470  2      DSA$V_Prompt = .Prompt;  ! Restore PROMPT flag
;    0471  2      DSA$V_Oper   = .Oper;  ! Restore OPER flag
;    0472  2
;    0473  2      !+
;    0474  2      ! Now use the ^C Menu Selection table entry chosen to decide what to do
```

```
;    0475  2      !-
;    0476  2
;    0477  2      CASE (.Entry_Ptr [CntlCTbl$B_Entry_CntlCSel]) FROM 1 TO (MEN$K_CntlC_Sel_Last_Entry - 1) OF
;    0478  2  SET
;    0479  2  [MEN$K_CntlC_Sel_Exit ]:
;    0480  2      !+
;    0481  2      ! Don't worry about resetting the DisablCC bit in the DS$GL_Flags longword because we are
;    0482  2      ! stopping CRD Menu Test here.
;    0483  2      !-
;    0484  2
;    0485  2      CRD$Stop (MENU$K_Exit_Operator_Abort);
;    0486  2
;    0487  2  [MEN$K_CntlC_Sel_Continue ]:
;    0488  2      !+
;    0489  2      ! Clear the CtrlC bit so that the CtrlC handler will not be called
;    0490  2      ! when enabled, due to ^C's being typed when CtrlC was disabled.
;    0491  2      ! Reset the DisablCC bit in the DS$GL_Flags longword so that ^C's can be typed again, and will
;    0492  2      ! be recognized by the VDS. Also, reset the handler addresses.
;    0493  2      !-
;    0494  2
;    0495  3      BEGIN
;    0496  3      $CRD_Message (New_Line, MEN$GT_CntlC_Continuing);
;    0497  3      CRD$Set_Ctrl_C_Handlers (MEN$CntlC_Menu, MEN$CntlC_Menu);
;    0498  3      CRD$Clear_Ctrl_C_Flag ();  ! Clear the DS$GL_Flags <DS$V_CtrlC> flag
;    0499  3      CRD$Enable_Ctrl_C ();
;    0500  3
;    0501  4      RETURN (CRD$K_Success)
;    0502  2      END;
;    0503  2
;    0504  2  [MEN$K_CntlC_Sel_FTMenu ]:
;    0505  2      !+
;    0506  2      ! Clear the CtrlC bit so that the CtrlC handler will not be called
;    0507  2      ! when enabled, due to ^C's being typed when CtrlC was disabled.
;    0508  2      ! Go back to the Main Menu. This will be accomplished by JSB'ing to the BEGIN_BLISS label
;    0509  2      ! in the KERNEL module. That code will reset the FP, fixup the stack pointer, etc., then
;    0510  2      ! it will call the DS$Cli routine. It, in turn, calls the DS_Superline routine which will
;    0511  2      ! print the DS> prompt, causing CRD to be called. This causes the Main Menu to be printed.
;    0512  2      !
;    0513  2      ! Reset the DisablCC bit in the DS$GL_Flags longword so that ^C's can be typed again, and will
;    0514  2      ! be recognized by the VDS. Also, reset the two ^C handler routine addresses.
;    0515  2      !-
;    0516  2
;    0517  3      BEGIN
;    0518  3      CRD$Set_Ctrl_C_Handlers (MEN$CntlC_Menu, MEN$CntlC_Menu);
;    0519  3      CRD$Clear_Ctrl_C_Flag (); ! Clear the DS$GL_Flags <DS$V_CtrlC> flag
;    0520  3
;    0521  3      CRD$GB_Current_State_Table = MEN$K_MM_State_Table;
;    0522  3      ! Set the current state table to the Main Menu
;    0523  3
;    0524  3      CRD$GL_MM_Current_State = MM$K_State_Print_Menu;
;    0525  3      ! Set the MM state to print the Main Menu
;    0526  3
;    0527  3      CRD$GL_TQ_Current_State = TQ$K_State_Init_TQ;
;    0528  3      ! Reset the TQ state number to it's initial state
;    0529  3
```

```
;    0530  3      CRD$GL_HS_Current_State = HS$K_State_Init_HS;
;    0531  3      ! Reset the HS state number to it's initial state
;    0532  3
;    0533  3      CRD$Enable_Ctrl_C (); ! Re-enable ^C' recognition
;    0534  3
;    0535  3      CRD$Screen_Pause (SCREEN$K_Clear_Screen);
;    0536  3      ! Clear screen before returning to Main Menu   [01]
;    0537  3
;    0538  3      JSB_Begin (.CRD$GA_Begin); ! JSB to the BEGIN label in the KERNEL module
;    0539  3      ! NOTE: This will not return to here!!!!
;    0540  4      RETURN (CRD$K_Success)
;    0541  2      END;
;    0542  2
;    0543  2  TES;
;    0544  2
;    0545  3     RETURN (CRD$K_Success)
;    0546  1 END;     ! Routine MEN$Cntlc_CMenu



            .EXTRN  CRD$DISABLE_CTRL_C
            .EXTRN  CRD$CLEAR_CTRL_C_HANDLERS
            .EXTRN  CRD$CLEAR_CTRL_C_FLAG
            .EXTRN  CRD$GB_USER_PROMPT_OKAY
            .EXTRN  MEN$GT_CNTLC_MENU_PROMPT
            .EXTRN  DS$BREAK, DS$ASKSTR
            .EXTRN  MEN$GT_INVOPERINPUT
            .EXTRN  CRD$STOP, MEN$GT_CNTLC_CONTINUING
            .EXTRN  CRD$SET_CTRL_C_HANDLERS
            .EXTRN  CRD$ENABLE_CTRL_C
            .EXTRN  CRD$GB_CURRENT_STATE_TABLE
            .EXTRN  CRD$GL_MM_CURRENT_STATE
            .EXTRN  CRD$GL_TQ_CURRENT_STATE
            .EXTRN  CRD$GL_HS_CURRENT_STATE
            .EXTRN  CRD$SCREEN_PAUSE
            .EXTRN  CRD$GA_BEGIN

              0FFC 00000     .ENTRY  MEN$CNTLC_MENU, Save R2,R3,R4,R5,R6,R7,R8,- ; 0297
             R9,R10,R11              ;
           5E     FF6C    CE  9E 00002     MOVAB   -148(SP), SP            ;
    00000000G     EF  16 00007     JSB     CRD$DISABLE_CTRL_C          ; 0414
    00000000G     EF  16 0000D     JSB     CRD$CLEAR_CTRL_C_HANDLERS         ; 0415
    00000000G     EF  16 00013     JSB     CRD$CLEAR_CTRL_C_FLAG           ; 0416
           58  D4 00019     CLRL    VALID_STATUS            ; 0422
           5E  DD 0001B 1$:   PUSHL   SP              ; 0426
      08   AE  9F 0001D     PUSHAB  CNTLC_TBL_PTR           ;
    FF2D   CF      02  FB 00020     CALLS   #2, TYPE_CONTROL_C_MENU         ;
    50 0000FE01    9F      01      05  EF 00025     EXTZV   #5, #1, a#^X0000FE01, R0        ; 0428
           56      50  90 0002E     MOVB    R0, PROMPT              ;
    50 0000FE01    9F      01      04  EF 00031     EXTZV   #4, #1, a#^X0000FE01, R0        ; 0429
           57      50  90 0003A     MOVB    R0, OPER                ;
    00000000G     EF      01  90 0003D     MOVB    #1, CRD$GB_USER_PROMPT_OKAY         ; 0431
    0000FE01      9F      30  88 00044     BISB2   #48, a#^X0000FE01           ; 0433
    00000000G     9F      00  FB 0004B     CALLS   #0, a#DS$BREAK          ; 0437
           7E  D4 00052     CLRL    -(SP)               ;
```

```
                00000000'  EF  9F 00054    PUSHAB  T_PROMPT_DEFAULT          ;
                    7E  D4 0005A    CLRL    -(SP)                ;
                    7E 48   8F  9A 0005C    MOVZBL  #72, -(SP)           ;
                20  AE  9F 00060    PUSHAB  OPERINPUT_BUFFER          ;
                 00000000G  EF  9F 00063    PUSHAB  MEN$GT_CNTLC_MENU_PROMPT          ;
                 00000000G  9F      06  FB 00069    CALLS   #6, @#DS$ASKSTR           ;
                    52      50  D0 00070    MOVL    R0, STATUS           ;
                 00000000G  9F      00  FB 00073    CALLS   #0, @#DS$BREAK           ;
                    4A      52  E9 0007A    BLBC    STATUS, 6$           ;
                    55      01  CE 0007D    MNEGL   #1, CNTLC_TBL_INDEX          ; 0440
                    3B  11 00080    BRB     5$                   ; 0449
                    50        55      06  C5 00082 2$:   MULL3   #6, CNTLC_TBL_INDEX, R0          ; 0445
                    54        50 04   AE  C1 00086    ADDL3   CNTLC_TBL_PTR, R0, ENTRY_PTR          ;
                08  AE      64  9A 0008B    MOVZBL  (ENTRY_PTR), VALID_ALPHA_UPPER          ; 0446
                0C  AE    08    AE      20  C1 0008F    ADDL3   #32, VALID_ALPHA_UPPER, VALID_ALPHA_LOWER    ; 0447
                    50 10   AE  9A 00095    MOVZBL  OPERINPUT_BUFFER, R0          ; 0449
                    5A  D4 00099    CLRL    R10                  ;
            50          00    08    AE      01  2D 0009B    CMPC5   #1, VALID_ALPHA_UPPER, #0, R0, -         ;
              11    AE      000A1    OPERINPUT_BUFFER+1           ;
                    02  12 000A3    BNEQ    3$                   ;
                    5A  D6 000A5    INCL    R10                  ;
                    50 10   AE  9A 000A7 3$:   MOVZBL  OPERINPUT_BUFFER, R0          ; 0450
                    59  D4 000AB    CLRL    R9                   ;
            50          00    0C    AE      01  2D 000AD    CMPC5   #1, VALID_ALPHA_LOWER, #0, R0, -         ;
              11    AE      000B3    OPERINPUT_BUFFER+1           ;
                    02  12 000B5    BNEQ    4$                   ;
                    59  D6 000B7    INCL    R9                   ;
                    58        59      5A  C9 000B9 4$:   BISL3   R10, R9, VALID_STATUS          ;
                    55  D6 000BD 5$:   INCL    CNTLC_TBL_INDEX          ; 0442
                    6E        55  D1 000BF    CMPL    CNTLC_TBL_INDEX, CNTLC_TBL_ENTRIES        ;
                    03  18 000C2    BGEQ    6$                   ;
                    BB        58  E9 000C4    BLBC    VALID_STATUS, 2$          ; 0443
                    40        58  E8 000C7 6$:   BLBS    VALID_STATUS, 8$          ; 0454
                    51 10   AE  9A 000CA    MOVZBL  OPERINPUT_BUFFER, R1          ; 0462
                    50 00000000'  EF  9A 000CE    MOVZBL  T_PROMPT_DEFAULT, R0          ; 0463
            50          00    11    AE      51  2D 000D5    CMPC5   R1, OPERINPUT_BUFFER+1, #0, R0, -        ;
               00000000'  EF      000DB    T_PROMPT_DEFAULT+1           ;
                    22  13 000E0    BEQL    7$                   ;
               00000000G  EF  9F 000E2    PUSHAB  CRD$GT_NEW_LINE_MESSAGE          ; 0465
                    01  DD 000E8    PUSHL   #1                   ;
                    1A  DD 000EA    PUSHL   #26                  ;
               00000000G  FF      03  FB 000EC    CALLS   #3, @#CRD$PRINT          ;
               00000000G  EF  9F 000F3    PUSHAB  MEN$GT_INVOPERINPUT          ;
                    01  DD 000F9    PUSHL   #1                   ;
                    1A  DD 000FB    PUSHL   #26                  ;
               00000000G  FF      03  FB 000FD    CALLS   #3, @#CRD$PRINT          ;
                    03        58  E8 00104 7$:   BLBS    VALID_STATUS, 8$          ; 0467
                  FF11  31 00107    BRW     1$                   ;
               00000000G  EF  94 0010A 8$:   CLRB    CRD$GB_USER_PROMPT_OKAY          ; 0469
0000FE01    9F          01        05      56  F0 00110    INSV    PROMPT, #5, #1, @#^X0000FE01         ; 0470
0000FE01    9F          01        04      57  F0 00119    INSV    OPER, #4, #1, @#^X0000FE01         ; 0471
                    02        01 01   A4  8F 00122    CASEB   1(ENTRY_PTR), #1, #2          ; 0477
                  0052      0012    0006      00127 9$:   .WORD   10$-9$,-           ;
                     11$-9$,-          ;
                     12$ 9$          ;
```

```
                0C  DD 0012D 10$:   PUSHL   #12             ; 0485
            00000000G   EF      01  FB 0012F    CALLS   #1, CRD$STOP          ;
              0086  31 00136    BRW     13$               ;
            00000000G   EF  9F 00139 11$:   PUSHAB  CRD$GT_NEW_LINE_MESSAGE        ; 0496
                01  DD 0013F    PUSHL   #1                ;
                1A  DD 00141    PUSHL   #26               ;
            00000000G   FF      03  FB 00143    CALLS   #3, @CRD$PRINT          ;
            00000000G   EF  9F 0014A    PUSHAB  MEN$GT_CNTLC_CONTINUING         ;
                01  DD 00150    PUSHL   #1                ;
                1A  DD 00152    PUSHL   #26               ;
            00000000G   FF      03  FB 00154    CALLS   #3, @CRD$PRINT          ;
                51      FEA1    CF  9E 0015B    MOVAB   MEN$CNTLC_MENU, R1          ; 0497
                50      FE9C    CF  9E 00160    MOVAB   MEN$CNTLC_MENU, R0          ;
            00000000G   EF  16 00165    JSB     CRD$SET_CTRL_C_HANDLERS         ;
            00000000G   EF  16 0016B    JSB     CRD$CLEAR_CTRL_C_FLAG           ; 0498
            00000000G   EF  16 00171    JSB     CRD$ENABLE_CTRL_C           ; 0499
                46  11 00177    BRB     13$               ; 0501
                51      FE83    CF  9E 00179 12$:   MOVAB   MEN$CNTLC_MENU, R1          ; 0518
                50      FE7E    CF  9E 0017E    MOVAB   MEN$CNTLC_MENU, R0          ;
            00000000G   EF  16 00183    JSB     CRD$SET_CTRL_C_HANDLERS         ;
            00000000G   EF  16 00189    JSB     CRD$CLEAR_CTRL_C_FLAG           ; 0519
            00000000G   EF  94 0018F    CLRB    CRD$GB_CURRENT_STATE_TABLE          ; 0521
            00000000G   EF      0F  D0 00195    MOVL    #15, CRD$GL_MM_CURRENT_STATE        ; 0524
            00000000G   EF      03  D0 0019C    MOVL    #3, CRD$GL_TQ_CURRENT_STATE         ; 0527
            00000000G   EF      03  D0 001A3    MOVL    #3, CRD$GL_HS_CURRENT_STATE         ; 0530
            00000000G   EF  16 001AA    JSB     CRD$ENABLE_CTRL_C           ; 0533
                01  DD 001B0    PUSHL   #1                ; 0535
            00000000G   EF      01  FB 001B2    CALLS   #1, CRD$SCREEN_PAUSE            ;
            00000000G   FF  16 001B9    JSB     @CRD$GA_BEGIN               ; 0538
                50      01  D0 001BF 13$:   MOVL    #1, R0              ; 0545
             04 001C2    RET                 ; 0546

; Routine Size:  451 bytes,    Routine Base:  CODE + 011A
```

```
;   0547  1 END
;   0548  0 ELUDOM



;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          11  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK          35  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE         733  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



Symbol                          Type  Defined  Referenced ...
--   -  ----- ------------ -- - ---- - - - - ------ --- ----
$ASCIC    Macro    Lib02   342
$CRD_LITERALS   Macro    Lib03    83
$CRD_MESSAGE   Macro    Lib03   371     396     404     465     496
$CRD_PRINT   Unbound          371     401     404     465     496
    Macro    Lib03   371     401     404     465     496
$DS_ASKSTR   KeyWMacr Lib02   437
$DS_BREAK   Macro    Lib02   437
$DS_DSADEF   Macro    Lib02    82
$DS_TYPEDEF   Macro    Lib02   371     401     404     465     496
$EQULST    Macro    Lib04    83      84      85     371     401     404     465     496
$ER    Comptime    88
$MEN_ASKSTR   KeyWMacr Lib03   435
$MEN_CNTLCTBL_ENTRY_ATTR Macro    Lib03   322
$MEN_CNTLCTBL_ENTRY_FLD  Fieldset Lib03   107     207     322     331     365
$MEN_LITERALS   Macro    Lib03    84
$MODULE    Bind         88
$SCREEN_LITERALS  Macro    Lib03    85
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     83
AUTO$K_EXIT_CONTROL_C  Literal     83
AUTO$K_EXIT_DUMMY_2  Literal     83
AUTO$K_EXIT_DUMMY_3  Literal     83
AUTO$K_EXIT_ERRORS_COMPLETE Literal     83
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     83
AUTO$K_EXIT_INTERNAL_ERROR Literal     83
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     83
AUTO$K_EXIT_LAST_REASON  Literal     83    83
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     83
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     83
AUTO$K_EXIT_NOERRORS_PREP Literal     83
CNTLCTBL$B_ENTRY_ALPHA  Field    Lib03   393a    445a    446.
CNTLCTBL$B_ENTRY_CNTLCSEL Field    Lib03   477.
CNTLCTBL$L_ENTRY_TEXT  Field    Lib03   181a    286a    394.
CNTLC_INITTBL_INIT  Own        157   157=    172.    186=
```

```
Symbol                          Type  Defined  Referenced ...
------ -------------------- -------- ----- ---------------
CNTLC_TBL_ENTRIES  Local       329   426a    442.
    Bind         361   381=    386=    391.
CNTLC_TBL_INDEX   Register    323   440=    442=    442.    445.
    Register    355   389=    391=    391.    393.    394.
CNTLC_TBL_INIT   Own         261   261=    276.    291=
CNTLC_TBL_PTR   Local       330   426a    445aa
    Bind         360   364m    380=    385=    393aa   394a.
CONTROL_C_TABLE_ENTRIES  RoutForm   352   361.
CONTROL_C_TABLE_PTR  RoutForm   352   360.
CRD$AL_CRD_DISPATCH_VECTORS External Lib03   183a    288a
CRD$CLEAR_CTRL_C_FLAG  ExtRout  Lib03   416c    498c    519c
CRD$CLEAR_CTRL_C_HANDLERS ExtRout  Lib03   415c
CRD$DISABLE_CTRL_C  ExtRout  Lib03   414c
CRD$ENABLE_CTRL_C  ExtRout  Lib03   499c    533c
CRD$GA_BEGIN   External Lib03   538ac
CRD$GB_CURRENT_STATE_TABLE External Lib03   521=
CRD$GB_USER_PROMPT_OKAY  External Lib03   431=    469-
CRD$GL_HS_CURRENT_STATE  External Lib03   530=
CRD$GL_MM_CURRENT_STATE  External Lib03   524=
CRD$GL_TQ_CURRENT_STATE  External Lib03   527=
CRD$GT_NEW_LINE_MESSAGE  External Lib03   371a    401a    404a    465a    496a
CRD$GT_STAR_LINE_PREFIX_MESSAGE Unbound          371     401     404     465     496
CRD$GT_STAR_LINE_SUFFIX_MESSAGE Unbound          371     401     404     465     496
CRD$K_ACTION_EQL_DO_NOTHING Literal     83
CRD$K_ACTION_RANGE_ERROR Literal     83
CRD$K_ADVANCE_DIAGNOSTIC Literal     83
CRD$K_ADVANCE_GENERAL_COM Literal     83
CRD$K_ADVANCE_STATE  Literal     83
CRD$K_ALLOCATION_ERROR  Literal     83
CRD$K_ASSIGN_PTABLE_PTRS Literal     83
CRD$K_AUTOSIZER_ERROR  Literal     83
CRD$K_AUTOSIZER_SET_FLAGS Literal     83
CRD$K_BAD_ACTION_NUMBER  Literal     83
CRD$K_BAD_TYPECODE_ERROR Literal     83
CRD$K_BASE_SYSTEM_IDC  Literal     83
CRD$K_BASE_SYSTEM_LESI  Literal     83
CRD$K_BASE_SYSTEM_NULL  Literal     83
CRD$K_BASE_SYSTEM_UDA_0  Literal     83
CRD$K_BASE_SYSTEM_UDA_1  Literal     83
CRD$K_BASE_SYSTEM_UDA_2  Literal     83
CRD$K_BASE_SYSTEM_UDA_3  Literal     83
CRD$K_CRD_INITIALIZATION Literal     83
CRD$K_CREATE_HST  Literal     83
CRD$K_DATA_LENGTH_ERROR  Literal     83
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     83
CRD$K_DDB_BLINK   Literal     83
CRD$K_DDB_FLINK   Literal     83
CRD$K_DDB_LAST_ENTRY  Literal     83
CRD$K_DDB_MEMORY_SIZE  Literal     83
CRD$K_DDB_MENU_DEVICE_NUMB Literal     83
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     83
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     83
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     83
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_DDB_PTABLE_ENTRY  Literal     83
CRD$K_DDB_STATUS  Literal     83
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     83
CRD$K_DEVICE_NAME  Literal     83
CRD$K_DIAGNOSTIC_ERROR  Literal     83
CRD$K_DIAGNOSTIC_NAME  Literal     83
CRD$K_DONE_GENERAL_COMS  Literal     83
CRD$K_DO_NOTHING  Literal     83
CRD$K_DUMMY_10   Literal     83
CRD$K_DUMMY_11   Literal     83
CRD$K_DUMMY_12   Literal     83
CRD$K_DUMMY_13   Literal     83
CRD$K_DUMMY_3   Literal     83
CRD$K_DUMMY_4   Literal     83
CRD$K_DUMMY_5   Literal     83
CRD$K_DUMMY_6   Literal     83
CRD$K_DUMMY_7   Literal     83
CRD$K_DUMMY_8   Literal     83
CRD$K_DUMMY_9   Literal     83
CRD$K_EXIT_CRD   Literal     83
CRD$K_HOOK_POINT  Literal     83
CRD$K_HS_INTERNAL_ERROR  Literal     83
CRD$K_IDC_EVDLB   Literal     83
CRD$K_IDC_EVDLC   Literal     83
CRD$K_IDC_EVDLD   Literal     83
CRD$K_IDC_EVRAD_0  Literal     83
CRD$K_IDC_EVRAD_1  Literal     83
CRD$K_IDC_LAST_DIAGNOSTIC Literal     83
CRD$K_INTERNAL_ERROR  Literal     83
CRD$K_LAST_ACTION_ROUTINE Literal     83
CRD$K_LAST_ENTRY  Literal     83
CRD$K_LAST_FUNCTION  Literal     83
CRD$K_LAST_HOOK_POINT  Literal     83
CRD$K_LAST_INTERNAL_ERROR Literal     83
CRD$K_LAST_TYPE   Literal     83
CRD$K_LESI_ENWCA  Literal     83
CRD$K_LESI_EVRMB_0  Literal     83
CRD$K_LESI_EVRMB_1  Literal     83
CRD$K_LESI_LAST_DIAGNOSTIC Literal     83
CRD$K_LEVEL_4_PASSED  Literal     83
CRD$K_LOAD_AUTOSIZER  Literal     83
CRD$K_LOAD_ERROR  Literal     83
CRD$K_LOAD_GENERAL_COM  Literal     83
CRD$K_LOAD_LOAD_COM  Literal     83
CRD$K_LOAD_SELECT_COM  Literal     83
CRD$K_LOAD_SET_EVENT_COM Literal     83
CRD$K_LOAD_SET_FLAGS_COM Literal     83
CRD$K_LOAD_START_COM  Literal     83
CRD$K_LOAD_SUP_FILE  Literal     83
CRD$K_MM_INTERNAL_ERROR  Literal     83
CRD$K_NEW_LINE   Literal     83
CRD$K_PREPARATION_ERROR  Literal     83
CRD$K_PREPARATION_ERROR_TEXT Literal     83
```

```
Symbol                            Type  Defined  Referenced ...
--------------------------------- ------- ----- ---------------
CRD$K_RUNTIME   Literal     83
CRD$K_SAME_LINE   Literal     83
CRD$K_SET_EVENT_ARGS  Literal     83
CRD$K_SET_FLAGS_ARGS  Literal     83
CRD$K_SET_UP_DIAGNOSTIC  Literal     83
CRD$K_START   Literal     83
CRD$K_START_AUTOSIZER  Literal     83
CRD$K_START_ERROR  Literal     83
CRD$K_START_QUALIFIERS  Literal     83
CRD$K_STAR_LINE   Literal     83
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal     83
CRD$K_STATE_ADVANCE_GENERAL_COM Literal     83
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal     83
CRD$K_STATE_AUTOSIZER_ERROR Literal     83
CRD$K_STATE_AUTOSIZER_RUNNING Literal     83
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal     83
CRD$K_STATE_DIAGNOSTIC_ERROR Literal     83
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal     83
CRD$K_STATE_DONE_DIAGNOSTICS Literal     83
CRD$K_STATE_DONE_GENERAL_COMS Literal     83
CRD$K_STATE_DUMMY_1  Literal     83
CRD$K_STATE_DUMMY_10  Literal     83
CRD$K_STATE_DUMMY_12  Literal     83
CRD$K_STATE_DUMMY_13  Literal     83
CRD$K_STATE_DUMMY_2  Literal     83
CRD$K_STATE_DUMMY_3  Literal     83
CRD$K_STATE_DUMMY_4  Literal     83
CRD$K_STATE_DUMMY_6  Literal     83
CRD$K_STATE_DUMMY_7  Literal     83
CRD$K_STATE_DUMMY_8  Literal     83
CRD$K_STATE_EXIT_CRD  Literal     83
CRD$K_STATE_INTERNAL_ERROR Literal     83
CRD$K_STATE_LAST_STATE  Literal     83
CRD$K_STATE_LEVEL_4_PASSED Literal     83
CRD$K_STATE_LOAD_AUTOSIZER Literal     83
CRD$K_STATE_LOAD_ERROR  Literal     83
CRD$K_STATE_LOAD_GENERAL_COM Literal     83
CRD$K_STATE_LOAD_LOAD_COM Literal     83
CRD$K_STATE_LOAD_SELECT_COM Literal     83
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     83
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     83
CRD$K_STATE_LOAD_START_COM Literal     83
CRD$K_STATE_PREPARATION_ERROR Literal     83
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     83
CRD$K_STATE_START  Literal     83
CRD$K_STATE_START_AUTOSIZER Literal     83
CRD$K_STATE_START_ERROR  Literal     83
CRD$K_SUCCESS   Literal  Lib03   188     293     501     540     545
CRD$K_TEST_START_TEXT  Literal     83
CRD$K_TQ_INTERNAL_ERROR  Literal     83
CRD$K_TYPECODE   Literal     83
CRD$K_UDA_0_EVDLB  Literal     83
CRD$K_UDA_0_EVDLC  Literal     83
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_UDA_0_EVDLD  Literal     83
CRD$K_UDA_0_EVRLA_0  Literal     83
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     83
CRD$K_UDA_1_EVDLB  Literal     83
CRD$K_UDA_1_EVDLC  Literal     83
CRD$K_UDA_1_EVDLD  Literal     83
CRD$K_UDA_1_EVRLA_0  Litera.     83
CRD$K_UDA_1_EVRLA_1  Literal     83
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     83
CRD$K_UDA_2_EVDLB  Literal     83
CRD$K_UDA_2_EVDLC  Literal     83
CRD$K_UDA_2_EVDLD  Literal     83
CRD$K_UDA_2_EVRLA_0  Literal     83
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     83
CRD$K_UDA_3_EVDLB  Literal     83
CRD$K_UDA_3_EVDLC  Literal     83
CRD$K_UDA_3_EVDLD  Literal     83
CRD$K_UDA_3_EVRLA_0  Literal     83
CRD$K_UDA_3_EVRLA_1  Literal     83
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     83
CRD$PRINT   External Lib03   371ac   401ac   404ac   465ac   496ac
CRD$SCREEN_PAUSE  ExtRout  Lib03   535c
CRD$SET_CTRL_C_HANDLERS  ExtRout  Lib03   497c    518c
CRD$STOP   ExtRout  Lib03   485c
DS$ASKSTR   ExtRout    437   437c
DS$BREAK   ExtRout    437   437c    437e
DS$K_PRINTB   Literal    371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_PRINTF   Literal    371   371
    Literal     371   371
    Literal     401   401
    Literal     401   401
    Literal     404   404
    Literal     404   404
    Literal     465   465
    Literal     465   465
    Literal     496   496
    Literal     496   496
DS$K_PRINTI   Literal    371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     465
    Literal     496
    Literal     496
DS$K_PRINTX   Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ABORT_PROGRAM  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ABORT_TEST  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_COMMAND_ERR  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_COMMAND_OUT  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DS$K_TYPE_CRD_AUTOTEST  Literal    371   371
    Literal     371    371
    Literal     401    401
    Literal     401    401
    Literal     404    404
    Literal     404    404
    Literal     465    465
    Literal     465    465
    Literal     496    496
    Literal     496    496
DS$K_TYPE_DS_PROMPT  Literal    371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_DS_START  Literal    371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ERRDEV  Literal    371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ERRHARD  Literal    371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ERROR_BODY  Literal    371
    Literal     371
    Literal     401
```

```
Symbol                            Type  Defined  Referenced ...
-------------------------------- -------- ----- ---------------
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ERROR_END  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ERRPREP  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ERRSOFT  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ERRSUP  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ERRSYS  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_ERR_HALT  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_EXCEPTION  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_EXCEPTION_HEAD Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_FIRST_PASS  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_GENERAL  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     496
DS$K_TYPE_GENERAL_ERROR  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_NO_TESTS  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_PARAM_ERROR  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_PROGRAM_END  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_PROGRAM_INFO  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_PROGRAM_START  Literal     371
    Literal     371
```

Symbol	Type	Defined	Referenced ...
	Literal	401	
	Literal	401	
	Literal	404	
	Literal	404	
	Literal	465	
	Literal	465	
	Literal	496	
	Literal	496	
DS$K_TYPE_QIO_INVADP	Literal	371	
	Literal	371	
	Literal	401	
	Literal	401	
	Literal	404	
	Literal	404	
	Literal	465	
	Literal	465	
	Literal	496	
	Literal	496	
DS$K_TYPE_QIO_NODRIVER	Literal	371	
	Literal	371	
	Literal	401	
	Literal	401	
	Literal	404	
	Literal	404	
	Literal	404	
	Literal	465	
	Literal	465	
	Literal	496	
	Literal	496	
DS$K_TYPE_QIO_WRONGVER	Literal	371	
	Literal	371	
	Literal	401	
	Literal	401	
	Literal	404	
	Literal	404	
	Literal	465	
	Literal	465	
	Literal	496	
	Literal	496	
DS$K_TYPE_SCRIPT_ECHO	Literal	371	
	Literal	371	
	Literal	401	
	Literal	401	
	Literal	404	
	Literal	404	
	Literal	465	
	Literal	465	
	Literal	496	
	Literal	496	
DS$K_TYPE_SCRIPT_PNF	Literal	371	
	Literal	371	
	Literal	401	
	Literal	401	
	Literal	404	

```
Symbol                            Type  Defined  Referenced ...
--------------------------------- ------- ----- ---------------
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_SCRIPT_PROMPT  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_SCRIPT_SKIP  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_SEQUENCE_ERROR Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_START_ERR  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_START_LIST  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     496
    Literal     496
DS$K_TYPE_SUMMARY  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DS$K_TYPE_USER_PROMPT  Literal     371
    Literal     371
    Literal     401
    Literal     401
    Literal     404
    Literal     404
    Literal     465
    Literal     465
    Literal     496
    Literal     496
DSA$AL_APTMAIL    Bind          82     82
DSA$GL_APTCOM     Bind          82
DSA$GL_DEVLEN     Bind          82
DSA$GL_DEVNAM     Bind          82
DSA$GL_ERRNO      Bind          82
DSA$GL_EVENT      Bind          82
DSA$GL_FLAGS      Bind          82   428      429      432      433      470      471
DSA$GL_MSGPTR     Bind          82
DSA$GL_MSGTYP     Bind          82
DSA$GL_PASSES     Bind          82
DSA$GL_PASSNO     Bind          82
DSA$GL_SECTNO     Bind          82
DSA$GL_SID    Bind          82
DSA$GL_SUBTNO     Bind          82
DSA$GL_TESTNO     Bind          82
DSA$GL_UNITS      Bind          82
DSA$V_OPER    Macro    Lib02    429      433      471
DSA$V_PROMPT    Macro    Lib02    428      432      470
ENTRY_PTR    Register    322    445=    446a.    477a.
FALSE    Literal  Lib03    157      261      422      469
GET1ST_     Macro    Lib04     83       84       85      371      401      404      465      496
GET2ND_     Unbound            83       84       85      371      401      404      465      496
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal      83
HS$K_ACTION_ADVANCE_GENERAL_COM Literal      83
HS$K_ACTION_ADVANCE_STATE Literal      83
HS$K_ACTION_CHANGE_TABLE Literal      83
HS$K_ACTION_DIAGNOSTIC_ERROR Literal      83
HS$K_ACTION_DONE_GENERAL_COMS Literal      83
HS$K_ACTION_DO_NOTHING  Literal      83
HS$K_ACTION_DUMMY_3  Literal      83
HS$K_ACTION_DUMMY_4  Literal      83
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_ACTION_DUMMY_5  Literal     83
HS$K_ACTION_DUMMY_6  Literal     83
HS$K_ACTION_INIT_HS  Literal     83
HS$K_ACTION_INTERNAL_ERROR Literal     83
HS$K_ACTION_LAST_ACTION  Literal     83
HS$K_ACTION_LOAD_ERROR  Literal     83
HS$K_ACTION_LOAD_GENERAL_COM Literal     83
HS$K_ACTION_LOAD_LOAD_COM Literal     83
HS$K_ACTION_LOAD_SELECT_COM Literal     83
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     83
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     83
HS$K_ACTION_LOAD_START_COM Literal     83
HS$K_ACTION_PREPARATION_ERROR Literal     83
HS$K_ACTION_START_ERROR  Literal     83
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     83
HS$K_STATE_ADVANCE_GENERAL_COM Literal     83
HS$K_STATE_CHANGE_TABLE  Literal     83
HS$K_STATE_DIAGNOSTIC_ERROR Literal     83
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     83
HS$K_STATE_DONE_GENERAL_COMS Literal     83
HS$K_STATE_DUMMY_1  Literal     83
HS$K_STATE_DUMMY_2  Literal     83
HS$K_STATE_DUMMY_3  Literal     83
HS$K_STATE_DUMMY_4  Literal     83
HS$K_STATE_DUMMY_6  Literal     83
HS$K_STATE_INIT_HS  Literal     83   530
HS$K_STATE_INTERNAL_ERROR Literal     83
HS$K_STATE_LAST_STATE  Literal     83
HS$K_STATE_LOAD_ERROR  Literal     83
HS$K_STATE_LOAD_GENERAL_COM Literal     83
HS$K_STATE_LOAD_LOAD_COM Literal     83
HS$K_STATE_LOAD_SELECT_COM Literal     83
HS$K_STATE_LOAD_SET_EVENT_COM Literal     83
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     83
HS$K_STATE_LOAD_START_COM Literal     83
HS$K_STATE_PREPARATION_ERROR Literal     83
HS$K_STATE_START_ERROR  Literal     83
INDEX    Register   154   178=    179=    179.    181.
    Local      258   283=    284=    284.    286.
JSB_BEGIN   Linkage  Lib03   538
MEN$CNTLCINIT_TBL_INIT  GlobRout   120 Lib03e     62f
MEN$CNTLC_MENU   GlobRout   297 Lib03e     59f    497a    518a
MEN$CNTLC_TBL_INIT  GlobRout   224 Lib03e     65f
MEN$GA_CNTLC_INITTBL  Own        105   108=    181a    380a
MEN$GA_CNTLC_TBL  Own        205   208=    286a    385a
MEN$GB_FNCTST_INIT  External Lib03   377.
MEN$GK_CNTLC_INITTBL_ENTRIES Literal    102   105     179     381
MEN$GK_CNTLC_TBL_ENTRIES Literal    202   205     284     386
MEN$GK_TEST_QUEUE  Literal     84
MEN$GT_CNTLCM_CHOICE  External Lib03   404a
MEN$GT_CNTLC_CONTINUING  External Lib03   496a
MEN$GT_CNTLC_MENU_PROMPT External Lib03   437a
MEN$GT_CNTLC_MENU_SELECTION External Lib03   401a
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$GT_CNTLC_MENU_TITLE  External Lib03   371a
MEN$GT_CNTLC_SEL_CONTINUE External Lib03   115a    219a
MEN$GT_CNTLC_SEL_EXIT  External Lib03   111a    211a
MEN$GT_CNTLC_SEL_FTMENU  External Lib03   215a
MEN$GT_INVOPERINPUT  External Lib03   465a
MEN$K_CNTLCTBL_ENTRY_SIZE Literal  Lib03   106     206     322     330     364
MEN$K_CNTLC_SEL_CONTINUE Literal     84   114     218     487
MEN$K_CNTLC_SEL_EXIT  Literal     84   110     210     479
MEN$K_CNTLC_SEL_FTMENU  Literal     84   214     504
MEN$K_CNTLC_SEL_LAST_ENTRY Literal     84   477
MEN$K_DEVTSTPREP_1  Literal     84
MEN$K_DEVTSTPREP_2  Literal     84
MEN$K_DEVTSTPREP_3  Literal     84
MEN$K_DEVTSTPREP_4  Literal     84
MEN$K_DEVTSTPREP_5  Literal     84
MEN$K_DEVTSTPREP_6  Literal     84
MEN$K_DEVTSTPREP_7  Literal     84
MEN$K_DEVTSTPREP_NULL  Literal     84
MEN$K_FTMRET_EXIT  Literal     84
MEN$K_FTMRET_FAILURE  Literal     84
MEN$K_FTMRET_MENU  Literal     84
MEN$K_FTMRET_TEST  Literal     84
MEN$K_FTMSEL_EXIT  Literal     84
MEN$K_FTMSEL_FNCTST_ALL  Literal     84
MEN$K_FTMSEL_LAST_ENTRY  Literal     84
MEN$K_FTMSEL_PREP  Literal     84
MEN$K_FTMSEL_SYSTEM_HDWR Literal     84
MEN$K_FTMSEL_TEST  Literal     84
MEN$K_HS_STATE_TABLE  Literal     83
MEN$K_MM_STATE_TABLE  Literal     83   521
MEN$K_PREP_ERROR_TEXT_LEN Literal     84
MEN$K_SELDEV_NUMB_START  Literal     84
MEN$K_TMENU_TSTALL_DEVNUMB Literal     84
MEN$K_TQ_STATE_TABLE  Literal     83
MEN$K_TSTCLA_ALL  Literal     84
MEN$K_TSTCLA_CARD  Literal     84
MEN$K_TSTCLA_COMM  Literal     84
MEN$K_TSTCLA_CPU  Literal     84
MEN$K_TSTCLA_DISK  Literal     84
MEN$K_TSTCLA_IO_ADAPTOR  Literal     84
MEN$K_TSTCLA_LAST_ENTRY  Literal     84
MEN$K_TSTCLA_MEMORY  Literal     84
MEN$K_TSTCLA_NULL  Literal     84
MEN$K_TSTCLA_PRINTER  Literal     84
MEN$K_TSTCLA_REAL  Literal     84
MEN$K_TSTCLA_TAPE  Literal     84
MEN$K_TSTCLA_TERM  Literal     84
MEN$K_TSTCTG_CPU  Literal     84
MEN$K_TSTCTG_IO_ADAPTOR  Literal     84
MEN$K_TSTCTG_IO_CONTROLLER Literal     84
MEN$K_TSTCTG_IO_UNIT  Literal     84
MEN$K_TSTCTG_MEMORY  Literal     84
MEN$K_TSTCTG_NULL  Literal     84
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$K_TSTTXT_DEVNAM_SZ  Literal     84
MEN$K_TSTTXT_TSTCLA_SZ  Literal     84
MEN$K_TSTTXT_UUTNAM_SZ  Literal     84
MENU$K_EXIT_AUTOSIZER_ERROR Literal     83
MENU$K_EXIT_INTERNAL_ERROR Literal     83
MENU$K_EXIT_LAST_REASON  Literal     83
MENU$K_EXIT_OPERATOR_ABORT Literal     83   485
MENU$K_EXIT_OPERATOR_STOP Literal     83
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     83
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     83
MM$K_ACTION_ADVANCE_STATE Literal     83
MM$K_ACTION_AUTOSIZER_ERROR Literal     83
MM$K_ACTION_BUILD_DDBS  Literal     83
MM$K_ACTION_BUILD_HSTS  Literal     83
MM$K_ACTION_CHANGE_TABLE Literal     83
MM$K_ACTION_DONE_INITS  Literal     83
MM$K_ACTION_DO_NOTHING  Literal     83
MM$K_ACTION_DUMMY_3  Literal     83
MM$K_ACTION_DUMMY_4  Literal     83
MM$K_ACTION_DUMMY_5  Literal     83
MM$K_ACTION_DUMMY_6  Literal     83
MM$K_ACTION_DUMMY_7  Literal     83
MM$K_ACTION_DUMMY_8  Literal     83
MM$K_ACTION_INTERNAL_ERROR Literal     83
MM$K_ACTION_LAST_ACTION  Literal     83
MM$K_ACTION_LOAD_AUTOSIZER Literal     83
MM$K_ACTION_MENU_PROMPT  Literal     83
MM$K_ACTION_PRINT_MENU  Literal     83
MM$K_ACTION_PRINT_MENU_HEADING Literal     83
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     83
MM$K_ACTION_START_AUTOSIZER Literal     83
MM$K_STATE_AUTOSIZER_ERROR Literal     83
MM$K_STATE_AUTOSIZER_RUNNING Literal     83
MM$K_STATE_BUILD_DDBS  Literal     83
MM$K_STATE_BUILD_HSTS  Literal     83
MM$K_STATE_CHANGE_TABLE  Literal     83
MM$K_STATE_DONE_INITS  Literal     83
MM$K_STATE_DUMMY_1  Literal     83
MM$K_STATE_DUMMY_2  Literal     83
MM$K_STATE_DUMMY_3  Literal     83
MM$K_STATE_DUMMY_5  Literal     83
MM$K_STATE_DUMMY_7  Literal     83
MM$K_STATE_DUMMY_8  Literal     83
MM$K_STATE_INTERNAL_ERROR Literal     83
MM$K_STATE_LAST_STATE  Literal     83
MM$K_STATE_LOAD_AUTOSIZER Literal     83
MM$K_STATE_MENU_PROMPT  Literal     83
MM$K_STATE_PRINT_MENU  Literal     83   524
MM$K_STATE_PRINT_MENU_HEADING Literal     83
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     83
MM$K_STATE_START  Literal     83
MM$K_STATE_START_AUTOSIZER Literal     83
MODNAM    Macro    L b01    88
```

```
Symbol                          Type  Defined  Referenced ...
------------------ ----------- ------- ----- ---------------
NUL2ND_     Macro    Lib04    83      84       85     371      401     404     465     496
ON     Literal  Lib03   432     433
OPER     Register   325   429=    471.
OPERINPUT_BUFFER  Local      339   437a    449.    450.    462.
OPERINPUT_BUFFER_SZ  Literal    336   339
PROMPT     Register   324   428=    470.
SCREEN$K_CLEAR_SCREEN  Literal     85   535
SCREEN$K_COUNT_LINE  Literal     85
SCREEN$K_PRESS_RETURN  Literal     85
SCREEN$K_PRESS_RETURN_MM Literal     85
SCREEN$K_PRESS_RETURN_TM Literal     85
SCREEN$K_TM_MSG   Literal     85
SELECT_ALPHA_PTR  Register   356   393=    401.
SELECT_TEXT_PTR   Register   357   394=    401.
STATUS     Local      437   437=    437.
TEXT_PTR   Register   153   181=    183a=   183a.
    Local      257   286=    288a=   288a.
TQ$K_ACTION_ADVANCE_STATE Literal     83
TQ$K_ACTION_CHANGE_TABLE Literal     83
TQ$K_ACTION_DONE_TEST_QUEUE Literal     83
TQ$K_ACTION_DO_NOTHING  Literal     83
TQ$K_ACTION_DUMMY_3  Literal     83
TQ$K_ACTION_DUMMY_4  Literal     83
TQ$K_ACTION_DUMMY_5  Literal     83
TQ$K_ACTION_GET_DDB  Literal     83
TQ$K_ACTION_INIT_TQ  Literal     83
TQ$K_ACTION_INTERNAL_ERROR Literal     83
TQ$K_ACTION_LAST_ACTION  Literal     83
TQ$K_STATE_CHANGE_TABLE  Literal     83
TQ$K_STATE_DONE_TEST_QUEUE Literal     83
TQ$K_STATE_DUMMY_1  Literal     83
TQ$K_STATE_DUMMY_2  Literal     83
TQ$K_STATE_DUMMY_3  Literal     83
TQ$K_STATE_DUMMY_4  Literal     83
TQ$K_STATE_DUMMY_5  Literal     83
TQ$K_STATE_GET_DDB  Literal     83
TQ$K_STATE_INIT_TQ  Literal     83   527
TQ$K_STATE_INTERNAL_ERROR Literal     83
TQ$K_STATE_LAST_STATE  Literal     83
TRUE     Literal  Lib03   186     291     431
TYPE CONTROL_C_MENU  Routine    352   426c
T_PROMPT_DEFAULT  Bind       342   437a    463.
VALID_ALPHA_LOWER  Local      333   447=    450.
VALID_ALPHA_UPPER  Local      332   446=    447.    449.
VALID_STATUS   Register   326   422=    443.    449=    454.    467.
```

```
  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]MENCNTLC.B32;1
    2  Module MENCNTLC
   49  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   51  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   53  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   55  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  548  Eludom MENCNTLC


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     a Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;          -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923          7          0          94          00:00.1
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486         48          9          62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0       1000          00:04.4




;     COMMAND QUALIFIERS

; BLISS MENCNTLC.B32/LIST=MENCNTLC.LIS/CROSS_REFERENCE/DEBUG/OBJECT=MENCNTLC.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  733 code + 46 data bytes
; Run Time:    00:42.9
; Elapsed Time:    01:22.2
; Lines/CPU Min:     766
; Lexemes/CPU-Min: 66792
```

```
; Memory Used: 309 pages
; Compilation Complete
```

```
;    0001  0
;    0002  0 MODULE MENGOA (MAIN=MEN$Generate_Online_Attach) =
;    0003  0
;    0004  0 !**************************************************************************
;    0005  0 !++
;    0006  0 ! COPYRIGHT (c) 1985
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    001C  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0 ! -
;    0024  0 !**************************************************************************
;    0025  0
```

```
;    0026  0
;    0027  1 BEGIN
;    0028  1
;    0029  1 !*****************************************************************
;    0030  1 !+
;    0031  1 !   Module libraries
;    0032  1 !-
;    0033  1 !*****************************************************************
;    0034  1
;    0035  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    0036  1 LIBRARY 'DISK$DS:[DS.EVSBB.DEF]EVSBB_DEF.L32';
;    0037  1
;    0038  1 !*****************************************************************
;    0039  1 !+
;    0040  1 !   $LITERAL Macro: This macro encodes three pieces of information
;    0041  1 !      into a long word as follows:
;    0042  1 !
;    0043  1 !    .BYTE The VMS Class code for this device
;    0044  1 !    .BYTE The VMS Type code for this device
;    0045  1 !       .WORD The two character name of the VSM driver, note anything
;    0046  1 !     larger than two characters is truncated to two characters
;    0047  1 !
;    0048  1 !   The resultant names appear as Item_<device_name>. This literal is
;    0049  1 !   compared to a computed longword. A match will cause the device to
;    0050  1 !   be attached.
;    0051  1 !-
;    0052  1 !*****************************************************************
;    0053  1
; M  0054  1 KEYWORDMACRO $LITERAL ( Name, Class, Type, Driver ) =
; M  0055  1     LITERAL
; M  0056  1  %NAME('Item ',Name) = Class+(Type)^8+(%ASCII%STRING(Driver))^16
;    0057  1 %;
```

```
;    0058  1
;    0059  1 !*****************************************************************
;    0060  1 !+
;    0061  1 !   $Attach Macro: This macro processes an FAO string creating an
;    0062  1 !   attach command. The command is then feed to VDS
;    0063  1 !   doing the actual attach.
;    0064  1 !-
;    0065  1 !*****************************************************************
;    0066  1
; M  0067  1 MACRO $ATTACH ( CNTRL ) [] =
; M  0068  1  BEGIN
; M  0069  1  FAO_Input_Dsc[0,00,16,0] = %CHARCOUNT ( CNTRL );
; M  0070  1  FAO_Input_Dsc[0,16,08,0] = DSC$K_DTYPE_T;
; M  0071  1  FAO_Input_Dsc[0,24,08,0] = DSC$K_CLASS_S;
; M  0072  1  FAO_Input_Dsc[1,00,32,0] = UPLIT (%ASCII CNTRL);
; M  0073  1
; M  0074  1  Status = $FAO (
; M  0075  1   FAO_Input_Dsc,
; M  0076  1   FAO_Length,
; M  0077  1   FAO_Output_Dsc
; M  0078  1       %IF NOT %NULL(%REMAINING) %THEN ,%REMAINING %FI
; M  0079  1   );
; M  0080  1  IF NOT (.Status) THEN RETURN 0;
; M  0081  1
; M  0082  1  CH$COPY ( .FAO_Length, CH$PTR(FAO_Buf), %C' ',
; M  0083  1   .CLI_Buffer_Length, CH$PTR(.CLI_Buffer_Address)
; M  0084  1   );
;    0085  1  END%;
```

```
;    0086  1 !**************************************************************
;    0087  1 !*
;    0088  1 !   Reasons for various things:
;    0089  1 !
;    0090  1 !   01 The TS11/TU80/TS05 are seen as device type DT$_TS11 under VMS.
;    0091  1 !    a nonunique situation.
;    0092  1 !
;    0093  1 !   02 The DMF32A is missing DC$_ symbol and the driver, TX, is non-
;    0094  1 !    unique.
;    0095  1 !
;    0096  1 !   03  The DMF32P is missing the DC$_ symbol but was put in the DC$_LP
;    0097  1 !    class as EVSBB_DRA always showed this up as LP class. It is
;    0098  1 !    after all a parallel transfer type device like a Line Printer.
;    0099  1 !
;    0100  1 !   04 The DMC11 is allows reported by VMS as DT$_DMC11 even when the
;    0101  1 !    device is actually a DMR11. Therefore the device type is non -
;    0102  1 !    unique.
;    0103  1 !
;    0104  1 !**************************************************************
```

```
;    0105  1
;    0106  1 !**************************************************************************
;    0107  1 !+
;    0108  1 !   This section contains literal definitions for the individual devices
;    0109  1 !   that may be attached online. The definitions take the form of
;    0110  1 !   ITEM_<device_name>. An example is ITEM_RP06.
;    0111  1 !
;    0112  1 !   The user must add a new literal definitions. Don't forget to add
;    0113  1 !   a corresponding attach string in the SELECTONE list.
;    0114  1 !-
;    0115  1 !**************************************************************************
;    0116  1
;    0117  1 $LITERAL ( Name=RC25, Class=DC$_DISK, Type=DT$_RZ01, Driver=DA );
;    0118  1 $LITERAL ( Name=RCF25, Class=DC$_DISK, Type=DT$_RZF01, Driver=DA );
;    0119  1
;    0120  1 $LITERAL ( Name=RP04, Class=DC$_DISK, Type=DT$_RP04, Driver=DB );
;    0121  1 $LITERAL ( Name=RP05, Class=DC$_DISK, Type=DT$_RP05, Driver=DB );
;    0122  1 $LITERAL ( Name=RP06, Class=DC$_DISK, Type=DT$_RP06, Driver=DB );
;    0123  1
;    0124  1 $LITERAL ( Name=TU58, Class=DC$_DISK, Type=DT$_TU58, Driver=DD );
;    0125  1
;    0126  1 $LITERAL ( Name=RA60, Class=DC$_DISK, Type=DT$_RA60, Driver=DJ );
;    0127  1 $LITERAL ( Name-RA80, Class-DC$_DISK, Type=DT$_RA80, Driver=DJ );
;    0128  1
;    0129  1 $LITERAL ( Name=RL01, Class=DC$_DISK, Type=DT$_RL01, Driver=DL );
;    0130  1 $LITERAL ( Name=RL02, Class=DC$_DISK, Type=DT$_RL02, Driver=DL );
;    0131  1
;    0132  1 $LITERAL ( Name=RK06, Class=DC$_DISK, Type=DT$_RK06, Driver=DM );
;    0133  1 $LITERAL ( Name=RK07, Class=DC$_DISK, Type=DT$_RK07, Driver=DM );
;    0134  1
;    0135  1 $LITERAL ( Name=RM03, Class=DC$_DISK, Type=DT$_RM03, Driver=DR );
;    0136  1 $LITERA  ( Name=RM05, Class=DC$_DISK, Type=DT$_RM05, Driver=DR );
;    0137  1 $LITERAL ( Name-RM80, Class=DC$_DISK, Type=DT$_RM80, Driver=DR );
;    0138  1 $LITERAL ( Name=RP07, Class=DC$_DISK, Type=DT$_RP07, Driver=DR );
;    0139  1
;    0140  1 $LITERAL ( Name=RX50, Class=DC$_DISK, Type=DT$_RX50, Driver=DU );
;    0141  1 $LITERAL ( Name-RA81, Class=DC$_DISK, Type=DT$_RA81, Driver=DU );
;    0142  1
;    0143  1 $LITERAL ( Name=RX02, Class=DC$_DISK, Type=DT$_RX02, Driver=DY );
;    0144  1
;    0145  1 $LITERAL ( Name-DMF32P, Class=DC$_LP, Type=DT$_DMF32, Driver=LC ); !R03
;    0146  1
;    0147  1 $LITERAL ( Name=TU78, Class=DC$_TAPE, Type=DT$_TU78, Driver=MF );
;    0148  1
;    0149  1 !LITERAL ( Name=TS05, Class=DC$_TAPE, Type=DT$_TS11,  Driver=MS ); !R01
;    0150  1 !LITERAL ( Name=TS11, Class=DC$_TAPE, Type=DT$_TS11, Driver=MS ); !R01
;    0151  1 !LITERAL ( Name=TU80, Class=DC$_TAPE, Type=DT$_TS11, Driver=MS ); !R01
;    0152  1
;    0153  1 $LITERAL ( Name=TE16, Class=DC$_TAPE, Type=DT$_TE16, Driver=MT );
;    0154  1 $LITERAL ( Name=TU45, Class=DC$_TAPE, Type=DT$_TU45, Driver=MT );
;    0155  1 $LITERAL ( Name-TU77, Class=DC$_TAPE, Type=DT$_TU77, Driver=MT );
;    0156  1
;    0157  1 $LITERAL ( Name-TK50, Class=DC$_TAPE, Type=DT$_TK50, Driver=MU );
;    0158  1 $LITERAL ( Name=TU81, Class=DC$_TAPE, Type-DT$_TU81, Driver=MU );
;    0159  1
```

```
;    0160   1 $LITERAL ( Name=DZ32, Class=DC$_TERM, Type=DT$_DZ32, Driver=TT );
;    0161   1
;    0162   1 !LITERAL ( Name=DHU11, Class=DC$_TERM, Type=DT$_DHU, Driver=TX ); !R02
;    0163   1 !LITERAL ( Name=DMF32A, Class=DC$_????, Type=DT$_DMF32, Driver=TX ); !R02
;    0164   1
;    0165   1 $LITERAL ( Name=VS100, Class=DC$_WORKSTATION, Type=DT$_VS100, Driver=VB );
;    0166   1 $LITERAL ( Name=VS125, Class=DC$_WORKSTATION, Type=DT$_VS125, Driver=VB );
;    0167   1 $LITERAL ( Name=VS300, Class=DC$_WORKSTATION, Type=DT$_VS300, Driver=VB );
;    0168   1
;    0169   1 $LITERAL ( Name=DR11W, Class=DC$_REALTIME, Type=DT$_DR11W, Driver=XA );
;    0170   1
;    0171   1 $LITERAL ( Name=UNA11, Class=DC$_SCOM, Type=DT$_UNA11, Driver=XE );
;    0172   1
;    0173   1 $LITERAL ( Name=DMF32S, Class=DC$_SCOM, Type=DT$_DMF32, Driver=XG );
;    0174   1
;    0175   1 !LITERAL ( Name=DMC11, Class=DC$_SCOM, Type=DT$_DMC11, Driver=XM ); !R04
;    0176   1 !LITERAL ( Name=DMR11, Class=DC$_SCOM, Type=DT$_DMR11, Driver=XM ); !R04
```

```
;    0177  1
;    0178  1 GLOBAL ROUTINE MEN$Generate_Online_Attach
;    0179  1   ( CLI_Buffer_Length, CLI_Buffer_Address) =
;    0180  1
;    0181  1 !*****************************************************************
;    0182  1 !+
;    0183  1 !   Function:
;    0184  1 ! This routine will look at the global record to attach the device
;    0185  1 !    specified in the record. Devices are attached using default
;    0186  1 !    attach strings. Should no matching attach string be found, no
;    0187  1 !    action is taken and a zero is returned in R0.
;    0188  1 !
;    0189  1 !    Also be aware that a zero returned can also mean an internal
;    0190  1 !    failure of some sort. Routines that are called always return
;    0191  1 !       a status. Should the status be bad a zero is returned from this
;    0192  1 !    routine.
;    0193  1 !
;    0194  1 !   Input:
;    0195  1 !    CRD$GA_Rec_Buf  - the EVSBB record address.
;    0196  1 ! The CLI buffer length and address values
;    0197  1 !
;    0198  1 !   Return status in R0:
;    0199  1 !    A zero indicates device not attacned.
;    0200  1 !    A one indicates device attached.
;    0201  1 !-
;    0202  1 !*****************************************************************
;    0203  1
;    0204  2 BEGIN
;    0205  2
;    0206  2 LITERAL
;    0207  2     FAO_BUF_SIZE = 80;
;    0208  2
;    0209  2 EXTERNAL
;    0210  2     CRD$GA_Rec_Buf : ADDRESSING_MODE (LONG_RELATIVE);
;    0211  2 MAP
;    0212  2     CRD$GA_Rec_Buf : BLOCK [,BYTE];
;    0213  2
;    0214  2 OWN
;    0215  2     FAO_BUF   : BLOCK [FAO_BUF_SIZE,BYTE],
;    0216  2     FAO_Input_Dsc : BLOCK [2,LONG],
;    0217  2     FAO_Length   : WORD,
;    0218  2     FAO_Output_DSC : BLOCK [2,LONG],
;    0219  2     Item   : LONG,
;    0220  2     Status   : LONG,
;    0221  2     T1,T2   : LONG;
;    0222  2
```

```
;    0223  2
;    0224  2 ! Test that device is on this CPU as oppossed to another CPU or HSC50.
;    0225  2
;    0226  2 IF (.CRD$GA_Rec_Buf[EVSBB$B_Whose] NEQ EVSBB$_Mine) THEN RETURN 0;
;    0227  2
;    0228  2 ! Set up the FAO output descriptor dynamically, can't be done at compile time
;    0229  2
;    0230  2 FAO_Output_Dsc[0,00,16,0] = FAO_Buf_Size;
;    0231  2 FAO_Output_Dsc[0,16,08,0] = DSC$K_DTYPE_T;
;    0232  2 FAO_Output_Dsc[0,24,08,0] = DSC$K_CLASS_S;
;    0233  2 FAO_Output_Dsc[1,00,32,0] = FAO_Buf;
;    0234  2
;    0235  2 ! Set up the comparsion longword from the input data
;    0236  2
;    0237  2 Item<00,08,0> = .CRD$GA_Rec_Buf[EVSBB$B_Class];
;    0238  2 Item<08,08,0> = .CRD$GA_Rec_Buf[EVSBB$B_Type];
;    0239  2 Item<16,16,0> = .(CRD$GA_Rec_Buf[EVSBB$T_NAME]+1);
;    0240  2
;    0241  2 ! Now do the appropriate thing for the device specified
;    0242  2
;    0243  2 SELECTONE .Item OF
;    0244  2     SET
;    0245  2 !*******************************************************************
;    0246  2      [ Item_DR11W   ] :
;    0247  2 !     .   ATTACH DR11W HUB <name><unit> <csr> <vec> <br>
; P  0248  2      $ATTACH ('ATTACH DR11W HUB !AC!UW 760000 300 05',
; P  0249  2     CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P  0250  2     .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;    0251  2    );
;    0252  2 !*******************************************************************
;    0253  2 !     [ Item_DMC11   ] :
;    0254  2 !     .   ATTACH DMC11 HUB <name><unit> <csr> <vec> <br>
;    0255  2 !     $ATTACH ('ATTACH DMC11 HUB !AC!UW 760000 300 05',
;    0256  2 !   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
;    0257  2 !   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;    0258  2 !   );
;    0259  2
;    0260  2 !*******************************************************************
;    0261  2 !     [ Item_DMR11   ] :
;    0262  2 !     .   ATTACH DMR11 HUB <name><unit> <csr> <vec> <br>
;    0263  2 !     $ATTACH ('ATTACH DMR11 HUB !AC!UW 760000 300 05',
;    0264  2 !   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
;    0265  2 !   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;    0266  2 !   );
;    0267  2
;    0268  2 !*******************************************************************
;    0269  2 !     [ Item DMF32A ] :
;    0270  2 !
;    0271  2 !     .   ATTACH DMF32A HUB <name> <csr> <vec> <br> -
;    0272  2 !     .     .   <active lines> <baud rate> <loopback> <unibus init jumper>
;    0273  2 ! $ATTACH ('ATTACH DMF32A HUB !AC! 760000 300 05 377 9600 INTERNAL YES',
;    0274  2 !   CRD$GA_Rec_Buf[EVSBB$T_NAME]  ! Device name
;    0275  2 !   );
;    0276  2
;    0277  2 !*******************************************************************
```

```
;    0278  2     [ Item_DMF32P ] :
;    0279  2
;    0280  2 !     .   ATTACH DMF32P HUB <name> <csr> <vec> <br> <port device>
; P  0281  2  $ATTACH ('ATTACH DMF32P HUB !AC 760000 300 05 OTHER',
; P  0282  2   CRD$GA_Rec_Buf[EVSBB$T_NAME]  ! Device name
;    0283  2   );
;    0284  2
;    0285  2 !*************************************************************************
;    0286  2     [ Item_DMF32S ] :
;    0287  2
;    0288  2 !     .   ATTACH DMF32S HUB <name><unit> <csr> <vec> <br> <external wrap>
; P  0289  2  $ATTACH ('ATTACH DMF32S HUB !AC!UW 760000 300 05 NONE',
; P  0290  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P  0291  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;    0292  2   );
;    0293  2
;    0294  2 !*************************************************************************
;    0295  2     [ Item_RA60  ] :
;    0296  2 !     .   ATTACH RA60 HUB <name><unit>
; P  0297  2     $ATTACH ('ATTACH RA60 HUB !AC!UW',
; P  0298  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P  0299  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;    0300  2   );
;    0301  2
;    0302  2 !*************************************************************************
;    0303  2     [ Item_RA80  ] :
;    0304  2 !     .   ATTACH RA80 HUB <name><unit>
; P  0305  2     $ATTACH ('ATTACH RA80 HUB !AC!UW',
; P  0306  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P  0307  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;    0308  2   );
;    0309  2
;    0310  2 !*************************************************************************
;    0311  2     [ Item_RA81  ] :
;    0312  2 !     .   ATTACH RA81 HUB <name><unit>
; P  0313  2     $ATTACH ('ATTACH RA81 HUB !AC!UW',
; P  0314  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P  0315  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Dev'ce unit number
;    0316  2   );
;    0317  2
;    0318  2 !*************************************************************************
;    0319  2     [ Item_RC25 ] :
;    0320  2
;    0321  2 !     .   ATTACH RC25 HUB <name>
;    0322  3  BEGIN
; P  0323  3  $ATTACH ('ATTACH RC25 HUB !AC!UW',
; P  0324  3   CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
; P  0325  3   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Unit number
;    0326  3   );
;    0327  2  END;
;    0328  2
;    0329  2 !*************************************************************************
;    0330  2     [ Item_RCF25 ] :
;    0331  2
;    0332  2 !     .   ATTACH RCF25 HUB <name>
```

```
;    0333  3  BEGIN
; P  0334  3  $ATTACH ('ATTACH RCF25 HUB !AC!UW',
; P  0335  3   CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
; P  0336  3   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Unit number
;    0337  3   );
;    0338  2  END;
;    0339  2
;    0340  2 !*************************************************************************
;    0341  2     [ Item_RK06 ] :
;    0342  2
;    0343  2 !      ATTACH RK06 HUB <name><unit>
; P  0344  2  $ATTACH ('ATTACH RK06 HUB .AC!UW',
; P  0345  2   CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Dev'ce name
; P  0346  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device un't number
;    0347  2   );
;    0348  2
;    0349  2 !*************************************************************************
;    0350  2     [ Item RK07 ] :
;    0351  2
;    0352  2 !      ATTACH RK07 HUB <name><unit>
; P  0353  2  $ATTACH ( 'ATTACH RK07 HUB !AC!UW',
; P  0354  2   CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
; P  0355  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;    0356  2   );
;    0357  2 !*************************************************************************
;    0358  2     [ Item_RL01 ] :
;    0359  2
;    0360  2 !      ATTACH RL01 HUB <name><unit>
; P  0361  2  $ATTACH ( 'ATTACH RL01 HUB !AC!UW',
; P  0362  2   CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
; P  0363  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device un't number
;    0364  2   );
;    0365  2 !*************************************************************************
;    0366  2     [ Item_RL02 ] :
;    0367  2
;    0368  2 !      ATTACH RL02 HUB <name><unit>
; P  0369  2  $ATTACH ( 'ATTACH RL02 HUB !AC!UW',
; P  0370  2   CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
; P  0371  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;    0372  2   );
;    0373  2
;    0374  2 !*************************************************************************
;    0375  2     [ Item_RM03 ] :
;    0376  2
;    0377  2 !      ATTACH RM03 HUB <name><unit>
; P  0378  2  $ATTACH ( 'ATTACH RM03 HUB !AC!UW',
; P  0379  2   CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
; P  0380  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;    0381  2   );
;    0382  2
;    0383  2 !*************************************************************************
;    0384  2     [ Item_RM05 ] :
;    0385  2
;    0386  2 !      ATTACH RM05 HUB <name><unit>
; P  0387  2  $ATTACH ( 'ATTACH RM05 HUB !AC!UW',
```

```
;P 0388  2    CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
;P 0389  2    .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;   0390  2    );
;   0391  2
;   0392  2 !*****************************************************************
;   0393  2      [ Item_RM80 ] :
;   0394  2
;   0395  2 !      .   ATTACH RM80 HUB <name><unit>
;P 0396  2    $ATTACH ( 'ATTACH RM80 HUB !AC!UW',
;P 0397  2     CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
;P 0398  2     .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;   0399  2     );
;   0400  2
;   0401  2 !*****************************************************************
;   0402  2      [ Item_RP04 ] :
;   0403  2
;   0404  2 !      .   ATTACH RP04 HUB <name><unit>
;P 0405  2    $ATTACH ( 'ATTACH RP04 HUB !AC!UW',
;P 0406  2     CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
;P 0407  2     .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;   0408  2     );
;   0409  2
;   0410  2 !*****************************************************************
;   0411  2      [ Item_RP05 ] :
;   0412  2
;   0413  2 !      .   ATTACH RP05 HUB <name><unit>
;P 0414  2    $ATTACH ( 'ATTACH RP05 HUB !AC!UW',
;P 0415  2     CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
;P 0416  2     .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;   0417  2     );
;   0418  2
;   0419  2 !*****************************************************************
;   0420  2      [ Item_RP06 ] :
;   0421  2
;   0422  2 !      .   ATTACH RP06 HUB <name><unit>
;P 0423  2    $ATTACH ( 'ATTACH RP06 HUB !AC!UW',
;P 0424  2     CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
;P 0425  2     .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;   0426  2     );
;   0427  2
;   0428  2 !*****************************************************************
;   0429  2      [ Item_RP07 ] :
;   0430  2
;   0431  2 !      .   ATTACH RP07 HUB <name><unit>
;P 0432  2    $ATTACH ( 'ATTACH RP07 HUB !AC!UW',
;P 0433  2     CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
;P 0434  2     .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;   0435  2     );
;   0436  2
;   0437  2 !*****************************************************************
;   0438  2      [ Item_RX02 ] :
;   0439  2
;   0440  2 !      .   ATTACH RX02 HUB <name><unit>
;P 0441  2    $ATTACH ( 'ATTACH RX02 HUB !AC!UW',
;P 0442  2     CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
```

```
;P 0443  2    .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;  0444  2    );
;  0445  2
;  0446  2 !*************************************************************************
;  0447  2      [ Item_RX50   ] :
;  0448  2 !      .   ATTACH RX50 HUB <name><unit>
;P 0449  2      $ATTACH ('ATTACH RX50 HUB !AC!UW',
;P 0450  2    CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
;P 0451  2    .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;  0452  2    );
;  0453  2
;  0454  2 !*************************************************************************
;  0455  2      [ Item_TE16   ] :
;  0456  2 !      .   ATTACH TE16 HUB <name><unit>
;P 0457  2      $ATTACH ('ATTACH VS100 HUB !AC!UW',
;P 0458  2    CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
;P 0459  2    .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;  0460  2    );
;  0461  2
;  0462  2 !*************************************************************************
;  0463  2      [ Item_TK50   ] :
;  0464  2 !      .   ATTACH TK50 HUB <name><unit> <csr> <vec> <br>
;P 0465  2      $ATTACH ('ATTACH TK50 HUB !AC!UW 760000 300 05',
;P 0466  2    CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
;P 0467  2    .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;  0468  2    );
;  0469  2
;  0470  2 !*************************************************************************
;  0471  2 !      [ Item_TS11   ] :
;  0472  2 !      .   ATTACH TS11 HUB <name><unit> <csr> <vec> <br>
;  0473  2 !      $ATTACH ('ATTACH TS11 HUB !AC!UW 760000 300 05',
;  0474  2 !    CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
;  0475  2 !    .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;  0476  2 !    );
;  0477  2
;  0478  2 !*************************************************************************
;  0479  2      [ Item_TU45   ] :
;  0480  2 !      .   ATTACH TU45 HUB <name><unit>
;P 0481  2      $ATTACH ('ATTACH TU45 HUB !AC!UW',
;P 0482  2    CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
;P 0483  2    .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;  0484  2    );
;  0485  2
;  0486  2 !*************************************************************************
;  0487  2      [ Item_TU58 ] :
;  0488  2
;  0489  2 !      .   ATTACH TU58 HUB <name><unit> <csr> <vec> <br>
;P 0490  2    $ATTACH ( 'ATTACH TU58 HUB !AC!UW 760000 300 05',
;P 0491  2    CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
;P 0492  2    .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;  0493  2    );
;  0494  2
;  0495  2 !*************************************************************************
;  0496  2      [ Item_TU77   ] :
;  0497  2 !      .   ATTACH TU77 HUB <name><unit>
```

```
; P 0498  2     $ATTACH ('ATTACH TU77 HUB !AC!UW',
; P 0499  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P 0500  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;   0501  2   );
;   0502  2
;   0503  2 !*****************************************************************
;   0504  2     [ Item_TU78  ] :
;   0505  2 !     .   ATTACH TU78 HUB <name><unit>
; P 0506  2     $ATTACH ('ATTACH TU78 HUB !AC!UW',
; P 0507  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P 0508  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;   0509  2   );
;   0510  2
;   0511  2 !*****************************************************************
;   0512  2 !    [ Item_TU80  ] :
;   0513  2 !     .   ATTACH TU80 HUB <name><unit> <csr> <vec> <br>
;   0514  2 !    $ATTACH ('ATTACH TU80 HUB !AC!UW 760000 300 05',
;   0515  2 !  CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
;   0516  2 !  .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;   0517  2 !  );
;   0518  2
;   0519  2 !*****************************************************************
;   0520  2     [ Item_TU81  ] :
;   0521  2 !     .   ATTACH TU81 HUB <name><unit> <csr> <vec> <br>
; P 0522  2     $ATTACH ('ATTACH TU81 HUB !AC!UW 760000 300 05',
; P 0523  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P 0524  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;   0525  2   );
;   0526  2
;   0527  2 !*****************************************************************
;   0528  2     [ Item_VS100  ] :
;   0529  2 !     .   ATTACH VS100 HUB <name><unit> <csr> <vec> <br>
; P 0530  2     $ATTACH ('ATTACH VS100 HUB !AC!UW 760000 300 05',
; P 0531  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P 0532  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;   0533  2   );
;   0534  2
;   0535  2 !*****************************************************************
;   0536  2     [ Item_VS125  ] :
;   0537  2 !     .   ATTACH VS125 HUB <name><unit> <csr> <vec> <br>
; P 0538  2     $ATTACH ('ATTACH VS125 HUB !AC!UW 760000 300 05',
; P 0539  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P 0540  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;   0541  2   );
;   0542  2
;   0543  2 !*****************************************************************
;   0544  2     [ Item_VS300  ] :
;   0545  2 !     .   ATTACH VS300 HUB <name><unit> <csr> <vec> <br>
; P 0546  2     $ATTACH ('ATTACH VS300 HUB !AC!UW 760000 300 05',
; P 0547  2   CRD$GA_Rec_Buf[EVSBB$T_NAME],  ! Device name
; P 0548  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT]  ! Device unit number
;   0549  2   );
;   0550  2
;   0551  2 !*****************************************************************
;   0552  2     [ Item_UNA11 ] :
```

```
;    0553  2
;    0554  2 !     .   ATTACH UNA11 HUB <name><unit> <csr> <vec> <br>
; P  0555  2  $ATTACH ( 'ATTACH UNA11 HUB !AC!UW 760000 300 05',
; P  0556  2   CRD$GA_Rec_Buf[EVSBB$T_NAME], ! Device name
; P  0557  2   .CRD$GA_Rec_Buf[EVSBB$W_UNIT] ! Device unit number
;    0558  2   );
;    0559  2
;    0560  2 !*************************************************************************
;    0561  2     [ OTHERWISE ] : RETURN 0 ! Nothing attacned
;    0562  2     TES;
;    0563  2
;    0564  2 RETURN 1;
;    0565  1 END;


            .TITLE  MENGOA

            .PSECT  $PLIT$,NOWRT,NOEXE,2

55  48  20  57  31  31  52  44  20  48  43  41  54  54  41  00000 P.AAA:  .ASCII  \ATTACH DR11W HUB .AC!UW 760000 300 05\<0>  ;
30  30  30  30  36  37  20  57  55  21  43  41  21  20  42  0000F            ;
        00  35  30  20  30  30  33  20  0001E                ;
            00  00  00026    .ASCII  <0><0>            ;
48  20  50  32  33  46  4D  44  20  48  43  41  54  54  41  00028 P.AAB:  .ASCII  \ATTACH DMF32P HUB !AC 760000 300 05 OTHE\  ;
33  20  30  30  30  30  36  37  20  43  41  21  20  42  55  00037            ;
      45  48  54  4F  20  35  30  20  30  30  00046            ;
          00  00  00  52  00050    .ASCII  \R\<0><0><0>        ;
48  20  53  32  33  46  4D  44  20  48  43  41  54  54  41  00054 P.AAC:  .ASCII  \ATTACH DMF32S HUB !AC!UW 760000 300 05 N\  ;
30  30  30  36  37  20  57  55  21  43  41  21  20  42  55  00063            ;
      4E  20  35  30  20  30  30  33  20  30  00072            ;
          00  45  4E  4F  0007C    .ASCII  \ONE\<0>            ;
42  55  48  20  30  36  41  52  20  48  43  41  54  54  41  00080 P.AAD:  .ASCII  \ATTACH RA60 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  0008F                ;
42  55  48  20  30  38  41  52  20  48  43  41  54  54  41  00098 P.AAE:  .ASCII  \ATTACH RA80 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  000A7                ;
42  55  48  20  31  38  41  52  20  48  43  41  54  54  41  000B0 P.AAF:  .ASCII  \ATTACH RA81 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  000BF                ;
42  55  48  20  35  32  43  52  20  48  43  41  54  54  41  000C8 P.AAG:  .ASCII  \ATTACH RC25 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  000D7                ;
55  48  20  35  32  46  43  52  20  48  43  41  54  54  41  000E0 P.AAH:  .ASCII  \ATTACH RCF25 HUB !AC!UW\<0>        ;
    00  57  55  21  43  41  21  20  42  000EF                ;
42  55  48  20  36  30  4B  52  20  48  43  41  54  54  41  000F8 P.AAI:  .ASCII  \ATTACH RK06 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  00107                ;
42  55  48  20  37  30  4B  52  20  48  43  41  54  54  41  00110 P.AAJ:  .ASCII  \ATTACH RK07 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  0011F                ;
42  55  48  20  31  30  4C  52  20  48  43  41  54  54  41  00128 P.AAK:  .ASCII  \ATTACH RL01 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  00137                ;
42  55  48  20  32  4F  4C  52  20  48  43  41  54  54  41  00140 P.AAL:  .ASCII  \ATTACH RL02 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  0014F                ;
42  55  48  20  33  30  4D  52  20  48  43  41  54  54  41  00158 P.AAM:  .ASCII  \ATTACH RM03 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  00167                ;
42  55  48  20  35  30  4D  52  20  48  43  41  54  54  41  00170 P.AAN:  .ASCII  \ATTACH RM05 HUB !AC.UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  0017F                ;
42  55  48  20  30  38  4D  52  20  48  43  41  54  54  41  00188 P.AAO:  .ASCII  \ATTACH RM80 HUB !AC!UW\<0><0>       .;
    00  00  57  55  21  43  41  21  20  00197                ;
```

```
42  55  48  20  34  30  50  52  20  48  43  41  54  54  41  001A0 P.AAP:  .ASCII  \ATTACH RP04 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  001AF                          ;
42  55  48  20  35  30  50  52  20  48  43  41  54  54  41  001B8 P.AAQ:  .ASCII  \ATTACH RP05 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  001C7                          ;
42  55  48  20  36  30  50  52  20  48  43  41  54  54  41  001D0 P.AAR:  .ASCII  \ATTACH RP06 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  001DF                          ;
42  55  48  20  37  30  50  52  20  48  43  41  54  54  41  001E8 P.AAS:  .ASCII  \ATTACH RP07 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  001F7                          ;
42  55  48  20  32  30  58  52  20  48  43  41  54  54  41  00200 P.AAT:  .ASCII  \ATTACH RX02 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  0020F                          ;
42  55  48  20  30  35  58  52  20  48  43  41  54  54  41  00218 P.AAU:  .ASCII  \ATTACH RX50 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  00227                          ;
55  48  20  30  30  31  53  56  20  48  43  41  54  54  41  00230 P.AAV:  .ASCII  \ATTACH VS100 HUB !AC!UW\<0>         ;
    00  57  55  21  43  41  21  20  42  0023F                          ;
42  55  48  20  30  35  4B  54  20  48  43  41  54  54  41  00248 P.AAW:  .ASCII  \ATTACH TK50 HUB !AC!UW 760000 300 05\      ;
20  30  30  30  30  36  37  20  57  55  21  43  41  21  20  00257                  ;
        35  30  20  30  30  33  00266                      ;
42  55  48  20  35  34  55  54  20  48  43  41  54  54  41  0026C P.AAX:  .ASCII  \ATTACH TU45 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  0027B                          ;
42  55  48  20  38  35  55  54  20  48  43  41  54  54  41  00284 P.AAY:  .ASCII  \ATTACH TU58 HUB !AC!UW 760000 300 05\      ;
20  30  30  30  30  36  37  20  57  55  21  43  41  21  20  00293                  ;
        35  30  20  30  30  33  002A2                      ;
42  55  48  20  37  37  55  54  20  48  43  41  54  54  41  002A8 P.AAZ:  .ASCII  \ATTACH TU77 HUB !AC.UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  002B7                          ;
42  55  48  20  38  37  55  54  20  48  43  41  54  54  41  002C0 P.ABA:  .ASCII  \ATTACH TU78 HUB !AC!UW\<0><0>       ;
    00  00  57  55  21  43  41  21  20  002CF                          ;
42  55  48  20  31  38  55  54  20  48  43  41  54  54  41  002D8 P.ABB:  .ASCII  \ATTACH TU81 HUB !AC!UW 760000 300 05\      ;
20  30  30  30  30  36  37  20  57  55  21  43  41  21  20  002E7                  ;
        35  30  20  30  30  33  002F6                      ;
55  48  20  30  30  31  53  56  20  48  43  41  54  54  41  002FC P.ABC:  .ASCII  \ATTACH VS100 HUB !AC!UW 760000 300 05\<0>  ;
30  30  30  30  36  37  20  57  55  21  43  41  21  20  42  0030B                  ;
        00  35  30  20  30  30  33  20  0031A                      ;
            00  00  00322     .ASCII  <0><0>                  ;
55  48  20  35  32  31  53  56  20  48  43  41  54  54  41  00324 P.ABD:  .ASCII  \ATTACH VS125 HUB !AC!UW 760000 300 05\<0>  ;
30  30  30  30  36  37  20  57  55  21  43  41  21  20  42  00333                  ;
        00  35  30  20  30  30  33  20  00342                      ;
            00  00  0034A     .ASCII  <0><0>                  ;
55  48  20  30  30  33  53  56  20  48  43  41  54  54  41  0034C P.ABE:  .ASCII  \ATTACH VS300 HUB !AC!UW 760000 300 05\<0>  ;
30  30  30  30  36  37  20  57  55  21  43  41  21  20  42  0035B                  ;
        00  35  30  20  30  30  33  20  0036A                      ;
            00  00  00372     .ASCII  <0><0>                  ;
55  48  20  31  31  41  4E  55  20  48  43  41  54  54  41  00374 P.ABF:  .ASCII  \ATTACH UNA11 HUB !AC UW 760000 300 05\<0>  ;
30  30  30  30  36  37  20  57  55  21  43  41  21  20  42  00383                  ;
        00  35  30  20  30  30  33  20  00392                      ;
            00  00  0039A     .ASCII  <0><0>                  ;

        .PSECT  $OWN$,NOEXE,2

        00000 FAO_BUF:.BLKB   80
        00050 FAO_INPUT_DSC:
        .BLKB   8
        00058 FAO_LENGTH:
        .BLKB   2
        0005A   .BLKB   2
        0005C FAO_OUTPUT_DSC:
```

```
                .BLKB   8
                00064 ITEM:     .BLKB   4
                00068 STATUS:   .BLKB   4
                0006C T1:       .BLKB   4
                00070 T2:       .BLKB   4

                .EXTRN  CRD$GA_REC_BUF, SYS$FAO

                .PSECT  $CODE$,NOWRT,2

                  007C 00000     .ENTRY  MEN$GENERATE_ONLINE_ATTACH, Save R2,R3,R4,- ; 0178
                 R5,R6            ;
                05 00000000G  EF  91 00002     CMPB    CRD$GA_REC_BUF+100, #5          ; 0226
                03  13 00009     BEQL    1$              ;
              03F3  31 0000B     BRW     35$             ;
0000'   CF 010E0050   8F  D0 0000E 1$:    MOVL    #17694800, FAO_OUTPUT_DSC           ; 0230
0000'   CF      0000' CF  9E 00017     MOVAB   FAO_BUF, FAO_OUTPUT_DSC+4           ; 0233
0000'   CF 00000000G EF  B0 0001E     MOVW    CRD$GA_REC_BUF+8, ITEM              ; 0237
0000'   CF 00000000G EF  B0 00027     MOVW    CRD$GA_REC_BUF+17, ITEM+2           ; 0239
        56      0000' CF  D0 00030     MOVL    ITEM, R6                ; 0243
   41580460     8F      56  D1 00035     CMPL    R6, #1096287328             ; 0246
        13  12 0003C     BNEQ    2$              ;
0000'   CF 010E0025   8F  D0 0003E     MOVL    #17694757, FAO_INPUT_DSC            ; 0251
0000'   CF      0000' CF  9E 00047     MOVAB   P.AAA, FAO_INPUT_DSC+4          ;
      0376  31 0004E     BRW     33$             ;
   434C0A43     8F      56  D1 00051 2$:    CMPL    R6, #1129056835             ; 0278
        2C  12 00058     BNEQ    3$              ;
0000'   CF 010E0029   8F  D0 0005A     MOVL    #17694761, FAO_INPUT_DSC            ; 0283
0000'   CF      0000' CF  9E 00063     MOVAB   P.AAB, FAO_INPUT_DSC+4          ;
   00000000G    EF  9F 0006A     PUSHAB  CRD$GA_REC_BUF+16           ;
        0000'   CF  9F 00070     PUSHAB  FAO_OUTPUT_DSC          ;
        0000'   CF  9F 00074     PUSHAB  FAO_LENGTH              ;
        0000'   CF  9F 00078     PUSHAB  FAO_INPUT_DSC           ;
   00000000G    00      04  FB 0007C     CALLS   #4, SYS$FAO             ;
        0361  31 00083     BRW     34$             ;
   47580A20     8F      56  D1 00086 3$:    CMPL    R6, #1196952096             ; 0286
        13  12 0008D     BNEQ    4$              ;
0000'   CF 010E002B   8F  D0 0008F     MOVL    #17694763, FAO_INPUT_DSC            ; 0292
0000'   CF      0000' CF  9E 00098     MOVAB   P.AAC, FAO_INPUT_DSC+4          ;
      0325  31 0009F     BRW     33$             ;
   4A441601     8F      56  D1 000A2 4$:    CMPL    R6, #1245976065             ; 0295
        13  12 000A9     BNEQ    5$              ;
0000'   CF 010E0016   8F  D0 000AB     MOVL    #17694742, FAO_INPUT_DSC            ; 0300
0000'   CF      0000' CF  9E 000B4     MOVAB   P.AAD, FAO_INPUT_DSC+4          ;
      0309  31 000BB     BRW     33$             ;
   4A441401     8F      56  D1 000BE 5$:    CMPL    R6, #1245975553             ; 0303
        13  12 000C5     BNEQ    6$              ;
0000'   CF 010E0016   8F  D0 000C7     MOVL    #17694742, FAO_INPUT_DSC            ; 0308
0000'   CF      0000' CF  9E 000D0     MOVAB   P.AAE, FAO_INPUT_DSC+4          ;
      02ED  31 000D7     BRW     33$             ;
   55441501     8F      56  D1 000DA 6$:    CMPL    R6, #1430525185             ; 0311
        13  12 000E1     BNEQ    7$              ;
0000'   CF 010E0016   8F  D0 000E3     MOVL    #17694742, FAO_INPUT_DSC            ; 0316
0000'   CF      0000' CF  9E 000EC     MOVAB   P.AAF, FAO_INPUT_DSC+4          ;
      02D1  31 000F3     BRW     33$             ;
```

```
      41441701   8F      56  D1 000F6 7$:   CMPL    R6, #1094981377          ; 0319
           13  12 000FD    BNEQ    8$             ;
0000'  CF 010E0016  8F  D0 000FF    MOVL    #17694742, FAO_INPUT_DSC          ; 0326
0000'  CF      0000' CF  9E 00108    MOVAB   P.AAG, FAO_INPUT_DSC+4          ;
      02B5  31 0010F    BRW     33$            ;
      41441801   8F      56  D1 00112 8$:   CMPL    R6, #1094981633          ; 0330
           13  12 00119    BNEQ    9$             ;
0000'  CF 010E0017  8F  D0 0011B    MOVL    #17694743, FAO_INPUT_DSC          ; 0337
0000'  CF      0000' CF  9E 00124    MOVAB   P.AAH, FAO_INPUT_DSC+4          ;
      0299  31 0012B    BRW     33$            ;
      4D440101   8F      56  D1 0012E 9$:   CMPL    R6, #1296302337          ; 0341
           13  12 00135    BNEQ    10$            ;
0000'  CF 010E0016  8F  D0 00137    MOVL    #17694742, FAO_INPUT_DSC          ; 0347
0000'  CF      0000' CF  9E 00140    MOVAB   P.AAI, FAO_INPUT_DSC+4          ;
      027D  31 00147    BRW     33$            ;
      4D440201   8F      56  D1 0014A 10$:   CMPL    R6, #1296302593          ; 0350
           13  12 00151    BNEQ    11$            ;
0000'  CF 010E0016  8F  D0 00153    MOVL    #17694742, FAO_INPUT_DSC          ; 0356
0000'  CF      0000' CF  9E 0015C    MOVAB   P.AAJ, FAO_INPUT_DSC+4          ;
      0261  31 00163    BRW     33$            ;
      4C440901   8F      56  D1 00166 11$:   CMPL    R6, #1279527169          ; 0358
           13  12 0016D    BNEQ    12$            ;
0000'  CF 010E0016  8F  D0 0016F    MOVL    #17694742, FAO_INPUT_DSC          ; 0364
0000'  CF      0000' CF  9E 00178    MOVAB   P.AAK, FAO_INPUT_DSC+4          ;
      0245  31 0017F    BRW     33$            ;
      4C440A01   8F      56  D1 00182 12$:   CMPL    R6, #1279527425          ; 0366
           13  12 00189    BNEQ    13$            ;
0000'  CF 010E0016  8F  D0 0018B    MOVL    #17694742, FAO_INPUT_DSC          ; 0372
0000'  CF      0000' CF  9E 00194    MOVAB   P.AAL, FAO_INPUT_DSC+4          ;
      0229  31 0019B    BRW     33$            ;
      52440601   8F      56  D1 0019E 13$:   CMPL    R6, #1380189697          ; 0375
           13  12 001A5    BNEQ    14$            ;
0000'  CF 010E0016  8F  D0 001A7    MOVL    #17694742, FAO_INPUT_DSC          ; 0381
0000'  CF      0000' CF  9E 001B0    MOVAB   P.AAM, FAO_INPUT_DSC+4          ;
      020D  31 001B7    BRW     33$            ;
      52440F01   8F      56  D1 001BA 14$:   CMPL    R6, #1380192001          ; 0384
           13  12 001C1    BNEQ    15$            ;
0000'  CF 010E0016  8F  D0 001C3    MOVL    #17694742, FAO_INPUT_DSC          ; 0390
0000'  CF      0000' CF  9E 001CC    MOVAB   P.AAN, FAO_INPUT_DSC+4          ;
      01F1  31 001D3    BRW     33$            ;
      52440D01   8F      56  D1 001D6 15$:   CMPL    R6, #1380191489          ; 0393
           13  12 001DD    BNEQ    16$            ;
0000'  CF 010E0016  8F  D0 001DF    MOVL    #17694742, FAO_INPUT_DSC          ; 0399
0000'  CF      0000' CF  9E 001E8    MOVAB   P.AAO, FAO_INPUT_DSC+4          ;
      01D5  31 001EF    BRW     33$            ;
      42440301   8F      56  D1 001F2 16$:   CMPL    R6, #1111753473          ; 0402
           13  12 001F9    BNEQ    17$            ;
0000'  CF 010E0016  8F  D0 001FB    MOVL    #17694742, FAO_INPUT_DSC          ; 0408
0000'  CF      0000' CF  9E 00204    MOVAB   P.AAP, FAO_INPUT_DSC+4          ;
      01B9  31 0020B    BRW     33$            ;
      42440401   8F      56  D1 0020E 17$:   CMPL    R6, #1111753729          ; 0411
           13  12 00215    BNEQ    18$            ;
0000'  CF 010E0016  8F  D0 00217    MOVL    #17694742, FAO_INPUT_DSC          ; 0417
0000'  CF      0000' CF  9E 00220    MOVAB   P.AAQ, FAO_INPUT_DSC+4          ;
      019D  31 00227    BRW     33$            ;
```

```
      42440501   8F      56  D1 0022A 18$:   CMPL    R6, #1111753985         ; 0420
          13  12 00231    BNEQ    19$             ;
0000'  CF 010E0016   8F  D0 00233    MOVL    #17694742, FAO_INPUT_DSC        ; 0426
0000'  CF      0000' CF  9E 0023C    MOVAB   P.AAR, FAO_INPUT_DSC+4           ;
       0181  31 00243    BRW     33$             ;
      52440701   8F      56  D1 00246 19$:   CMPL    R6, #1380189953         ; 0429
          13  12 0024D    BNEQ    20$             ;
0000'  CF 010E0016   8F  D0 0024F    MOVL    #17694742, FAO_INPUT_DSC        ; 0435
0000'  CF      0000' CF  9E 00258    MOVAB   P.AAS, FAO_INPUT_DSC+4           ;
       0165  31 0025F    BRW     33$             ;
      59440B01   8F      56  D1 00262 20$:   CMPL    R6, #1497631489         ; 0438
          13  12 00269    BNEQ    21$             ;
0000'  CF 010E0016   8F  D0 0026B    MOVL    #17694742, FAO_INPUT_DSC        ; 0444
0000'  CF      0000' CF  9E 00274    MOVAB   P.AAT, FAO_INPUT_DSC+4           ;
       0149  31 0027B    BRW     33$             ;
      55441A01   8F      56  D1 0027E 21$:   CMPL    R6, #1430526465         ; 0447
          13  12 00285    BNEQ    22$             ;
0000'  CF 010E0016   8F  D0 00287    MOVL    #17694742, FAO_INPUT_DSC        ; 0452
0000'  CF      0000' CF  9E 00290    MOVAB   P.AAU, FAO_INPUT_DSC+4           ;
       012D  31 00297    BRW     33$             ;
      544D0102   8F      56  D1 0029A 22$:   CMPL    R6, #1414332674         ; 0455
          13  12 002A1    BNEQ    23$             ;
0000'  CF 010E0017   8F  D0 002A3    MOVL    #17694743, FAO_INPUT_DSC        ; 0460
0000'  CF      0000' CF  9E 002AC    MOVAB   P.AAV, FAO_INPUT_DSC+4           ;
       0111  31 002B3    BRW     33$             ;
      554D0A02   8F      56  D1 002B6 23$:   CMPL    R6, #1431112194         ; 0463
          13  12 002BD    BNEQ    24$             ;
0000'  CF 010E0024   8F  D0 002BF    MOVL    #17694756, FAO_INPUT_DSC        ; 0468
0000'  CF      0000' CF  9E 002C8    MOVAB   P.AAW, FAO_INPUT_DSC+4           ;
       00F5  31 002CF    BRW     33$             ;
     544D0202   8F      56  D1 002D2 24$:   CMPL    R6, #1414332930         ; 0479
          13  12 002D9    BNEQ    25$             ;
0000'  CF 010E0016   8F  D0 002DB    MOVL    #17694742, FAO_INPUT_DSC        ; 0484
0000'  CF      0000' CF  9E 002E4    MOVAB   P.AAX, FAO_INPUT_DSC+4           ;
       00D9  31 002EB    BRW     33$             ;
     44440E01   8F      56  D1 002EE 25$:   CMPL    R6, #1145310721         ; 0487
          13  12 002F5    BNEQ    26$             ;
0000'  CF 010E0024   8F  D0 002F7    MOVL    #17694756, FAO_INPUT_DSC        ; 0493
0000'  CF      0000' CF  9E 00300    MOVAB   P.AAY, FAO_INPUT_DSC+4           ;
       00BD  31 00307    BRW     33$             ;
     544D0302   8F      56  D1 0030A 26$:   CMPL    R6, #1414333186         ; 0496
          13  12 00311    BNEQ    27$             ;
0000'  CF 010E0016   8F  D0 00313    MOVL    #17694742, FAO_INPUT_DSC        ; 0501
0000'  CF      0000' CF  9E 0031C    MOVAB   P.AAZ, FAO_INPUT_DSC+4           ;
       00A1  31 00323    BRW     33$             ;
     464D0502   8F      56  D1 00326 27$:   CMPL    R6, #1179452674         ; 0504
          13  12 0032D    BNEQ    28$             ;
0000'  CF 010E0016   8F  D0 0032F    MOVL    #17694742, FAO_INPUT_DSC        ; 0509
0000'  CF      0000' CF  9E 00338    MOVAB   P.ABA, FAO_INPUT_DSC+4           ;
       0085  31 0033F    BRW     33$             ;
     554D0802   8F      56  D1 00342 28$:   CMPL    R6, #1431111682         ; 0520
          12  12 00349    BNEQ    29$             ;
0000'  CF 010E0024   8F  D0 0034B    MOVL    #17694756, FAO_INPUT_DSC        ; 0525
0000'  CF      0000' CF  9E 00354    MOVAB   P.ABB, FAO_INPUT_DSC+4           ;
          6A  11 0035B    BRB     33$             ;
```

```
                42560146   8F      56  D1 0035D 29$:   CMPL    R6, #1112932678         ; 0528
                    12  12 00364    BNEQ    30$             ;
0000'  CF 010E0025   8F  D0 00366    MOVL    #17694757, FAO_INPUT_DSC        ; 0533
0000'  CF      0000' CF  9E 0036F    MOVAB   P.ABC, FAO_INPUT_DSC+4          ;
                    4F  11 00376    BRB     33$             ;
                42560246   8F      56  D1 00378 30$:   CMPL    R6, #1112932934         ; 0536
                    12  12 0037F    BNEQ    31$             ;
0000'  CF 010E0025   8F  D0 00381    MOVL    #17694757, FAO_INPUT_DSC        ; 0541
0000'  CF      0000' CF  9E 0038A    MOVAB   P.ABD, FAO_INPUT_DSC+4          ;
                    34  11 00391    BRB     33$             ;
                42560346   8F      56  D1 00393 31$:   CMPL    R6, #1112933190         ; 0544
                    12  12 0039A    BNEQ    32$             ;
0000'  CF 010E0025   8F  D0 0039C    MOVL    #17694757, FAO_INPUT_DSC        ; 0549
0000'  CF      0000' CF  9E 003A5    MOVAB   P.ABE, FAO_INPUT_DSC+4          ;
                    19  11 003AC    BRB     33$             ;
                45580E20   8F      56  D1 003AE 32$:   CMPL    R6, #1163398688         ; 0552
                    4A  12 003B5    BNEQ    35$             ;
0000'  CF 010E0025   8F  D0 003B7    MOVL    #17694757, FAO_INPUT_DSC        ; 0558
0000'  CF      0000' CF  9E 003C0    MOVAB   P.ABF, FAO_INPUT_DSC+4          ;
                7E 00000000G  EF  3C 003C7 33$:   MOVZWL  CRD$GA_REC_BUF+32, -(SP)       ;
                00000000G  EF  9F 003CE    PUSHAB  CRD$GA_REC_BUF+16       ;
                    0000'  CF  9F 003D4    PUSHAB  FAO_OUTPUT_DSC          ;
                    0000'  CF  9F 003D8    PUSHAB  FAO_LENGTH              ;
                    0000'  CF  9F 003DC    PUSHAB  FAO_INPUT_DSC           ;
                00000000G  00      05  FB 003E0    CALLS   #5, SYS$FAO             ;
0000'  CF       50  D0 003E7 34$:   MOVL    R0, STATUS              ;
                10      0000'  CF  E9 003EC    BLBC    STATUS, 35$             ;
            04  AC          20 0000'  CF     0000'  CF  2C 003F1    MOVC5   FAO_LENGTH, FAO_BUF, #32, -      ;
            08  BC      003FB    CLI_BUFFER_LENGTH, @CLI_BUFFER_ADDRESS      ;
                50       01  D0 003FD    MOVL    #1, R0                  ; 0564
              04 00400    RET                 ;
                50  D4 00401 35$:   CLRL    R0              ; 0565
              04 00403    RET                 ;

; Routine Size:  1028 bytes,    Routine Base:  $CODE$ + 0000
```

```
;   0566  1 END
;   0567  0 ELUDOM



;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  $OWN$          116  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$         924  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$        1028  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



Symbol                      Type  Defined  Referenced ...
--- - --- -------------- -------- ----- ----------------
$ATTACH      Macro       67   248     281     289     297     305     313     323     334     344     353     361
         369     378     387     396     405     414     423     432     441     449     457
         465     481     490     498     506     522     530     538     546     555
$FAO      Unbound           74
      Macro    Lib01   251     283     292     300     308     316     326     337     347     356     364
         372     381     390     399     408     417     426     435     444     452     460
         468     484     493     501     509     525     533     541     549     558
$LITERAL     KeyWMacr    54   117     118     120     121     122     124     126     127     129     130     132
         133     135     136     137     138     140     141     143     145     147     153
         154     155     157     158     160     165     166     167     169     171     173
CLASS       KeyWForm    54    56
CLI_BUFFER_ADDRESS    Unbound           83
      RoutForm   179   251a=   283a=   292a=   300a=   308a=   316a=   326a=   337a-   347a=   356a=   364a=
         372a=   381a=   390a=   399a=   408a=   417a=   426a=   435a=   444a=   452a=   460a=
         468a=   484a=   493a=   501a=   509a=   525a=   533a=   541a=   549a=   558a=
CLI_BUFFER_LENGTH    Unbound           83
      RoutForm   179   251.    283.    292.    300.    308.    316.    326.    337.    347.    356.    364.
         372.    381.    390.    399.    408.    417.    426.    435.    444.    452.    460.
         468.    484.    493.    501.    509.    525.    533.    541.    549.    558.
CNTRL       MacrForm    67    69       72
CRD$GA_REC_BUF     External   210   212m    226.    237.    238.    239.    251a    251.    283a    292a    292.    300a
         300.    308a    308.    316a    316.    326a    326.    337a    337.    347a    347.
         356a    356.    364a    364.    372a    372.    381a    381.    390a    390.    399a
         399.    408a    408.    417a    417.    426a    426.    435a    435.    444a    444.
         452a    452.    460a    460.    468a    468.    484a    484.    493a    493.    501a
         501.    509a    509.    525a    525.    533a    533.    541a    541.    549a    549.
         558a    558.
DC$_DISK     Literal  Lib01   117     118     120     121     122     124     126     127     129     130     132
         133     135     136     137     138     140     141     143
DC$_LP       Literal  Lib01   145
DC$_REALTIME     Literal  Lib01   169
DC$_SCOM     Literal  Lib01   171     173
DC$_TAPE     Literal  Lib01   147     153     154     155     157     158
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DC$_TERM        Literal  Lib01   160
DC$_WORKSTATION      Literal  Lib01   165     166     167
DRIVER       KeyWForm    54    56
DSC$K_CLASS_S      Unbound           71
       Literal  Lib01   232     251     283     292     300     308     316     326     337     347     356
           364     372     381     390     399     408     417     426     435     444     452
           460     468     484     493     501     509     525     533     541     549     558
DSC$K_DTYPE_T      Unbound           70
       Literal  Lib01   231     251     283     292     300     308     316     326     337     347     356
           364     372     381     390     399     408     417     426     435     444     452
           460     468     484     493     501     509     525     533     541     549     558
DT$_DMF32       Literal  Lib01   145     173
DT$_DR11W       Literal  Lib01   169
DT$_DZ32        Literal  Lib01   160
DT$_RA60        Literal  Lib01   126
DT$_RA80        Literal  Lib01   127
DT$_RA81        Literal  Lib01   141
DT$_RK06        Literal  Lib01   132
DT$_RK07        Literal  Lib01   133
DT$_RL01        Literal  Lib01   129
DT$_RL02        Literal  Lib01   130
DT$_RM03        Literal  Lib01   135
DT$_RM05        Literal  Lib01   136
DT$_RM80        Literal  Lib01   137
DT$_RP04        Literal  Lib01   120
DT$_RP05        Literal  Lib01   121
DT$_RP06        Literal  Lib01   122
DT$_RP07        Literal  Lib01   138
DT$_RX02        Literal  Lib01   143
DT$_RX50        Literal  Lib01   140
DT$_RZ01        Literal  Lib01   117
DT$_RZF01       Literal  Lib01   118
DT$_TE16        Literal  Lib01   153
DT$_TK50        Literal  Lib01   157
DT$_TU45        Literal  Lib01   154
DT$_TU58        Literal  Lib01   124
DT$_TU77        Literal  Lib01   155
DT$_TU78        Literal  Lib01   147
DT$_TU81        Literal  Lib01   158
DT$_UNA11       Literal  Lib01   171
DT$_VS100       Literal  Lib01   165
DT$_VS125       Literal  Lib01   166
DT$_VS300       Literal  Lib01   167
EVSBB$B_CLASS       Macro    Lib02   237
EVSBB$B_TYPE      Macro    Lib02   238
EVSBB$B_WHOSE       Macro    Lib02   226
EVSBB$T_NAME      Macro    Lib02   239     249     282     290     298     306     314     324     335     345     354
           362     370     379     388     397     406     415     424     433     442     450
           458     466     482     491     499     507     523     531     539     547     556
EVSBB$W_UNIT      Macro    Lib02   250     291     299     307     315     325     336     346     355     363     371
           380     389     398     407     416     425     434     443     451     459     467
           483     492     500     508     524     532     540     548     557
EVSBB$ MINE     Literal  Lib02   226
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
FAO_BUF       Unbound             82
      Own          215   233a    251.    283.    292.    300.    308.    316.    326.    337.    347.    356.
          364.     372.    381.    390.    399.    408.    417.    426.    435.    444.    452.
          460.     468.    484.    493.    501.    509.    525.    533.    541.    549.    558.
FAO_BUF_SIZE      Literal     207   215     230
FAO_INPUT_DSC      Unbound             69      70       71       72       75
      Own          216   251=    251a    283=    283a    292=    292a    300=    300a    308=    308a    316=
          316a     326=    326a    337=    337a    347=    347a    356=    356a    364=    364a
          372=     372a    381=    381a    390=    390a    399=    399a    408=    408a    417=
          417a     426=    426a    435=    435a    444-    444a    452=    452a    460=    460a
          468=     468a    484=    484a    493=    493a    501-    501a    509=    509a    525=
          525a     533=    533a    541=    541a    549=    549a    558=    558a
FAO_LENGTH      Unbound             76      82
      Own          217   251a    251.    283a    283.    292a    292.    300a    300.    308a    308.    316a
          316.     326a    326.    337a    337.    347a    347.    356a    356.    364a    364.
          372a     372.    381a    381.    390a    390.    399a    399.    408a    408.    417a
          417.     426a    426.    435a    435.    444a    444.    452a    452.    460a    460.
          468a     468.    484a    484.    493a    493.    501a    501.    509a    509.    525a
          525.     533a    533.    541a    541.    549a    549.    558a    558.
FAO_OUTPUT_DSC      Unbound             77
      Own          218   230=    231=    232=    233=    251a    283a    292a    300a    308a    316a    326a
          337a     347a    356a    364a    372a    381a    390a    399a    408a    417a    426a
          435a     444a    452a    460a    468a    484a    493a    501a    509a    525a    533a
          541a     549a    558a
ITEM        Own          219   237=    238=    239=    243.
ITEM_DMF32P      Literal     145   278
ITEM_DMF32S      Literal     173   286
ITEM_DR11W      Literal     169   246
ITEM_DZ32       Literal     160
ITEM_RA60       Literal     126   295
ITEM_RA80       Literal     127   303
ITEM_RA81       Literal     141   311
ITEM_RC25       Literal     117   319
ITEM_RCF25      Literal     118   330
ITEM_RK06       Literal     132   341
ITEM_RK07       Literal     133   350
ITEM_RL01       Literal     129   358
ITEM_RL02       Literal     130   366
ITEM_RM03       Literal     135   375
ITEM_RM05       Literal     136   384
ITEM_RM80       Literal     137   393
ITEM_RP04       Literal     120   402
ITEM_RP05       Literal     121   411
ITEM_RP06       Literal     122   420
ITEM_RP07       Literal     138   429
ITEM_RX02       Literal     143   438
ITEM_RX50       Literal     140   447
ITEM_TE16       Literal     153   455
ITEM_TK50       Literal     157   463
ITEM_TU45       Literal     154   479
ITEM_TU58       Literal     124   487
ITEM_TU77       Literal     155   496
ITEM_TU78       Literal     147   504
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
ITEM_TU81       Literal    158   520
ITEM_UNA11      Literal    171   552
ITEM_VS100      Literal    165   528
ITEM_VS125      Literal    166   536
ITEM_VS300      Literal    167   544
MEN$GENERATE_ONLINE_ATTACH GlobRout   179
NAME       KeyWForm    54    56
STATUS       Unbound             74       80
       Own           220   251=    251.    283=    283.    292=    292.    300=    300.    308=    308.    316=
           316.      326=    326.    337=    337.    347=    347.    356=    356.    364=    364.
           372=      372.    381=    381.    390=    390.    399=    399.    408=    408.    417=
           417.      426=    426.    435=    435.    444=    444.    452=    452.    460=    460.
           468=      468.    484=    484.    493=    493.    501=    501.    509=    509.    525=
           525.      533=    533.    541=    541.    549=    549.    558=    558.
SYS$FAO        ExtRout    251    251c    283e    283c    292e    292c    300e    300c    308e    308c    316e    316c
           326e      326c    337e    337c    347e    347c    356e    356c    364e    364c    372e
           372c      381e    381c    390e    390c    399e    399c    408e    408c    417e    417c
           426e      426c    435e    435c    444e    444c    452e    452c    460e    460c    468e
           468c      484e    484c    493e    493c    501e    501c    509e    509c    525e    525c
           533e      533c    541e    541c    549e    549c    558e    558c
T1         Own           221
T2         Own           221
TYPE       KeyWForm    54    56
```

CROSS REFERENCE MAP

```
Line #      Event  File ...
--- -    ---- --------      -----
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]MENGOA.B32;1
    2  Module MENGOA
   35  Library #1 SYS$COMMON:[SYSLIB]LIB.L32;2
   36  Library #2 DISK$DIAGPACK03:[DS.EVSBB.DEF]EVSBB_DEF.L32;1
  567  Eludom MENGOA
```

KEY TO REFERENCE TYPE FLAGS

```
     . Fetch
     = Store
     c Routine call
     a Address use
     @ Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
```

```
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  SYS$COMMON:[SYSLIB]LIB.L32;2       18619         42          0      1000           00:01.3
;  DISK$DIAGPACK03:[DS.EVSBB.DEF]EVSBB_DEF.L32;1
;               44          6         13          7          00:00.1




;     COMMAND QUALIFIERS

; BLISS MENGOA.B32/LIST=MENGOA.LIS/CROSS_REFERENCE/DEBUG/OBJECT=MENGOA.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  1028 code + 1040 data bytes
; Run Time:    00:27.9
; Elapsed Time:    00:49.6
; Lines/CPU Min:    1220
; Lexemes/CPU-Min: 36889
; Memory Used: 599 pages
; Compilation Complete
```

```
;    0001  0 %TITLE  'CRD MENU - Menu Prompt Module'
;    0002  0 MODULE MENPROMPT  (
;    0003  0   IDENT = 'V01.1'
;    0004  0   ) =
;    0005  0 !
;    0006  0 ! COPYRIGHT (C)  1982
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0 !
```

```
;    0024  0 !++
;    0025  0 ! FACILITY:
;    0026  0 !
;    0027  0 ! ABSTRACT:
;    0028  0 !
;    0029  0 ! ENVIRONMENT:
;    0030  0 !
;    0031  0 ! AUTHOR:
;    0032  0 !
;    0033  0 ! John Ciukaj  May-1982
;    0034  0 !
;    0035  0 ! CREATION DATE:
;    0036  0 !
;    0037  0 ! VERSION:
;    0038  0 !
;    0039  0 ! Modified by:
;    0040  0 !
;    0041  0 ! [01] John Ciukaj Version 1.1 Jan. 1983
;    0042  0 !
;    0043  0 !  - Added 'Test All' selection in Functional Test
;    0044  0 !    Menu Table
;    0045  0 !
;    0046  0 ! [02] Scott Cohen Version 1.2 June 17, 1983
;    0047  0 !  For Soft Console support
;    0048  0 !  - Added $SCREEN_Literals
;    0049  0 !  - Changed MEN$FuncTest_Menu to clear screen after operator input
;    0050  0 !  - Changed MEN$Print_System_Hdwr to pause after screen is full
;    0051  0 !
;    0052  0 ! [03] Ron Parker November 1984
;    0053  0 !  Added prep message #7
;    0054  0 ! -
```

```
;    0055  0 xSBTTL 'Libraries'
;    0056  1 BEGIN
;    0057  1
;    0058  1 LIBRARY '$DS';
;    0059  1 LIBRARY '$DIAG';
;    0060  1 LIBRARY '$CRD';
;    0061  1 LIBRARY 'SYS$LIBRARY:LIB';
;    0062  1
;    0063  1
```

```
;    0064  1 %SBTTL 'Forward-External  Routines'
;    0065  1
;    0066  1 FORWARD ROUTINE
;    0067  1     MEN$FuncTest_Menu    : ADDRESSING_MODE (LONG_RELATIVE),
;    0068  1     MEN$FTMTbl_Init    : ADDRESSING_MODE (LONG_RELATIVE),
;    0069  1     MEN$Test_Menu    : ADDRESSING_MODE (LONG_RELATIVE),
;    0070  1     MEN$PrepTbl_Init    : ADDRESSING_MODE (LONG_RELATIVE),
;    0071  1     MEN$Get_TstCla_Name  : ADDRESSING_MODE (LONG_RELATIVE),
;    0072  1     MEN$FndDDB_Tcl_SDN    : ADDRESSING_MODE (LONG_RELATIVE),
;    0073  1     MEN$FndDDB_Tcl    : ADDRESSING_MODE (LONG_RELATIVE),
;    0074  1     MEN$Print_DevTstPrep  : ADDRESSING_MODE (LONG_RELATIVE),
;    0075  1     MEN$Print_System_Hdwr  : ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    0076  1 %SBTTL 'Psect Definitions'
;    0077  1
;    0078  1 PSECT
;    0079  1     Plit = Data (NoExecute, Share, NoWrite, Addressing_Mode (Long_Relative));
;    0080  1 PSECT
;    0081  1     Code =  Code (Execute, Share, NoWrite, Addressing_Mode (Long_Relative), PIC);
;    0082  1 PSECT
;    0083  1     Own  = Work (NoExecute, NoShare, Write, Addressing_Mode (Long_Relative));
;    0084  1 PSECT
;    0085  1     Global = Work (NoExecute, NoShare, Write, Addressing_Mode (Long_Relat'ive));
```

```
;    0086  1 %SBTTL 'Macro and Literal Definitions'
;    0087  1
;    0088  1 $CRD_Literals;
;    0089  1 $MEN_Literals;
;    0090  1 $SCREEN_Literals;
```

```
;    0091  1 %SBTTL 'Module - Global Static Storage'
;    0092  1
;    0093  1      Modnam ('MENPROMPT');    ! Module Name for Errsup Macro
```

```
;    0094  1 %SBTTL 'Own Storage - Plits'
;    0095  1
;    0096  1 LITERAL
;    0097  1     HdwrNam_String_Buf_Sz = MEN$K_OperInput_Buffer_Size;
;    0098  1
;    0099  1 GLOBAL
;    0100  1     MEN$GT_OperInput_Buffer : $MEN_OperInput_Buffer_ATTR;
;    0101  1
;    0102  1 BIND
;    0103  1     HdwrNam_String_Buf = MEN$GT_OperInput_Buffer;
;    0104  1
;    0105  1 BIND
;    0106  1     T_Prompt_Default = $ASCIC ('?') : VECTOR [,BYTE];
```

```
;    0107  1 %SBTTL 'FUNCTIONAL TEST MAIN MENU Table'
;    0108  1
;    0109  1 !+
;    0110  1 ! Contains information relevant to all selections available
;    0111  1 ! from the Functional Test Menu.
;    0112  1 ! Because the CRD MENU code is loadable it is necessary to invoke the
;    0113  1 ! Initialization routine associated with this table following loading
;    0114  1 ! and before the table is accessed (see MENTSTINI module).
;    0115  1 !-
;    0116  1
;    0117  1 GLOBAL
;    0118  1     MEN$GA_FTMTbl : $MEN_FTMTbl_ATTR
;    0119  1       INITIAL (
;    0120  1
;    0121  1   BYTE (%C'A'),
;    0122  1   BYTE (MEN$K_FTMSel_Exit),
;    0123  1   LONG (MEN$GT_FTMSel_Exit),
;    0124  1
;    0125  1   BYTE (%C'B'),
;    0126  1   BYTE (MEN$K_FTMSel_System_Hdwr),
;    0127  1   LONG (MEN$GT_FTMSel_System_Hdwr),
;    0128  1
;    0129  1   BYTE (%C'C'),
;    0130  1   BYTE (MEN$K_FTMSel_Prep),
;    0131  1   LONG (MEN$GT_FTMSel_Prep),
;    0132  1
;    0133  1   BYTE (%C'D'),
;    0134  1   BYTE (MEN$K_FTMSel_Test),
;    0135  1   LONG (MEN$GT_FTMSel_Test),
;    0136  1
;    0137  1   BYTE (%C'E'),
;    0138  1   BYTE (MEN$K_FTMSel_FncTst_All),
;    0139  1   LONG (MEN$GT_FTMSel_FncTst_All)
;    0140  1   );
```

```
;    0141  1 %SBTTL 'FUNCTIONAL TEST MAIN MENU Table Initialization Routine'
;    0142  1
;    0143  1 GLOBAL ROUTINE MEN$FTMTbl_Init =
;    0144  1
;    0145  1 !++
;    0146  1 ! FUNCTIONAL DESCRIPTION:
;    0147  1 !
;    0148  1 !    Since the CRD MENU TEST control code is a loadable image,
;    0149  1 !    link-time resolved address pointers must have an offset added
;    0150  1 !    to them.  This is because the address pointers are resolved
;    0151  1 !    at link time with reference to a zero base address.  However,
;    0152  1 !    at run time the code will be loaded at an address starting at
;    0153  1 !    CRD$AL_CRD_Dispatch_Vectors.  Therefore, for all tables with text
;    0154  1 !    address pointers, add an offset equal to
;    0155  1 !    the value represented by the address location
;    0156  1 !    CRD$AL_CRD_Dispatch_Vectors!
;    0157  1 !
;    0158  1 ! INPUT PARAMETERS
;    0159  1 !
;    0160  1 ! OUTPUT PARAMETERS
;    0161  1 !
;    0162  1 ! IMPLICIT INPUTS
;    0163  1 !
;    0164  1 ! EXPLICIT OUTPUTS
;    0165  1 !
;    0166  1 ! SIDE EFFECTS
;    0167  1 !
;    0168  1 !    FTMTbl_Init is specified True.
;    0169  1 !
;    0170  1 ! COMPLETION CODES
;    0171  1 !
;    0172  1 !    CRD$K_Success
;    0173  1 !-
```

```
;    0174  2 BEGIN
;    0175  2     LOCAL
;    0176  2  Text_Ptr : REF VECTOR [, LONG],
;    0177  2  Index;
;    0178  2
;    0179  2     OWN
;    0180  2         FTMTbl_Init : BYTE INITIAL (BYTE(False));
;    0181  2      ! Flag location indicating whether
;    0182  2      ! Table Text Address pointers
;    0183  2      ! have been initialized with
;    0184  2      ! offset address  values to account
;    0185  2      ! for the fact that the code
;    0186  2      ! is loaded at a dynamic address
;    0187  2      ! location.
;    0188  2      ! 1 or True  = Table Initialized
;    0189  2      ! 0 or False = Table not initialized
;    0190  2
;    0191  2     !+
;    0192  2     ! Perform initialization only once
;    0193  2     !-
;    0194  2
;    0195  3     IF (NOT .FTMTbl_Init)
;    0196  2     THEN
;    0197  3  BEGIN
;    0198  3  !+
;    0199  3  ! Initialize Text Address Pointers for the FUNCTIONAL TEST Menu Table
;    0200  3  !-
;    0201  3
;    0202  3  Index = -1;
;    0203  3
;    0204  3  WHILE ((Index = .Index + 1) LSS (MEN$GK_FTMTbl_Entries)) DO
;    0205  4      BEGIN
;    0206  4      Text_Ptr = MEN$GA_FTMTbl [.Index, FTMTbl$L_Entry_Text];
;    0207  4      ! Address of text pointer
;    0208  4      Text_Ptr [0] = .Text_Ptr [0] + CRD$AL_CRD_Dispatch_Vectors;
;    0209  4      ! Add offset
;    0210  3      END;
;    0211  3  FTMTbl_Init = True;  ! Indicate that Init done
;    0212  2  END;
;    0213  2
;    0214  2     RETURN (CRD$K_Success);
;    0215  1 END;


        .TITLE  MENPROMPT CRD MENU - Menu Prompt Module
        .IDENT  \V01.1\

        .PSECT  WORK,NOEXE,2

        00000 MEN$GT_OPERINPUT_BUFFER::
        .BLKB   256
    41  00100 MEN$GA_FTMTBL::
        .BYTE   65              ;
    01  00101     .BYTE   1            ;
     00000000G 00102     .ADDRESS MEN$GT_FTMSEL_EXIT          ;
```

```
       42  00106     .BYTE   66              ;
       04  00107     .BYTE   4               ;
        00000000G 00108     .ADDRESS MEN$GT_FTMSEL_SYSTEM_HDWR        ;
       43  0010C     .BYTE   67              ;
       02  0010D     .BYTE   2               ;
        00000000G 0010E     .ADDRESS MEN$GT_FTMSEL_PREP          ;
       44  00112     .BYTE   68              ;
       03  00113     .BYTE   3               ;
        00000000G 00114     .ADDRESS MEN$GT_FTMSEL_TEST          ;
       45  00118     .BYTE   69              ;
       05  00119     .BYTE   5               ;
        00000000G 0011A     .ADDRESS MEN$GT_FTMSEL_FNCTST_ALL         ;
       00  0011E FTMTBL_INIT:
           .BYTE   0               ;

           .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      54  50  4D  4F  52  50  4E  45  4D  09  00000 P.AAA:  .ASCII  <9>\MENPROMPT\          ;
          3F  01  0000A P.AAB:  .ASCII  <1>\?\            ;

           $MODULE=        P.AAA
           HDWRNAM_STRING_BUF= MEN$GT_OPERINPUT_BUFFER
           T_PROMPT_DEFAULT=   P.AAB
            .EXTRN  MEN$FUNCTEST_MENU
            .EXTRN  MEN$FTMTBL_INIT
            .EXTRN  MEN$TEST_MENU, MEN$PREPTBL_INIT
            .EXTRN  MEN$GET_TSTCLA_NAME
            .EXTRN  MEN$FNDDDB_TCL_SDN
            .EXTRN  MEN$FNDDDB_TCL, MEN$PRINT_DEVTSTPREP
            .EXTRN  MEN$PRINT_SYSTEM_HDWR
            .EXTRN  MEN$GT_FTMSEL_EXIT
            .EXTRN  MEN$GT_FTMSEL_SYSTEM_HDWR
            .EXTRN  MEN$GT_FTMSEL_PREP
            .EXTRN  MEN$GT_FTMSEL_TEST
            .EXTRN  MEN$GT_FTMSEL_FNCTST_ALL
            .EXTRN  CRD$AL_CRD_DISPATCH_VECTORS

           .PSECT  CODE,NOWRT,  SHR,  PIC,2

             0004 00000     .ENTRY  MEN$FTMTBL_INIT, Save R2         ; 0143
           29 00000000'  EF  E8 00002    BLBS    FTMTBL_INIT, 3$          ; 0195
           50       01  CE 00009    MNEGL   #1, INDEX           ; 0202
           16  11 0000C    BRB     2$              ; 0204
           51        50      06  C5 0000E 1$:   MULL3   #6, INDEX, R1           ; 0206
           52 00000000'EF41  9E 00012    MOVAB   MEN$GA_FTMTBL+2[R1], TEXT_PTR         ;
           51 00000000G  EF  9E 0001A    MOVAB   CRD$AL_CRD_DISPATCH_VECTORS, R1        ; 0208
           62       51  C0 00021    ADDL2   R1, (TEXT_PTR)          ;
           50  D6 00024 2$:   INCL    INDEX           ; 0204
           05       50  D1 00026    CMPL    INDEX, #5           ;
           E3  19 00029    BLSS    1$              ;
      00000000'  EF       01  90 0002B    MOVB    #1, FTMTBL_INIT          ; 0211
           50       01  D0 00032 3$:   MOVL    #1, R0          ; 0214
            04 00035    RET                 ; 0215

; Routine Size:  54 bytes,    Routine Base:  CODE + 0000
```

```
;    0216  1 %SBTTL 'TEST MENU  Table'
;    0217  1
;    0218  1 !+
;    0219  1 ! Contains information relevant to all selections available
;    0220  1 ! from the Test Menu.
;    0221  1 !-
;    0222  1
;    0223  1 GLOBAL
;    0224  1     MEN$GA_TMTbl : $MEN_TMTbl_ATTR
;    0225  1      INITIAL (
;    0226  1  BYTE (0),   ! Selection Alpha.  This location
;    0227  1      ! is dynamically loaded at runtime.
;    0228  1      ! 0 = Invalid Test Menu selection
;    0229  1      ! Alpha A-Z = Valid Test Menu selection
;    0230  1  BYTE (MEN$K_TstCla_CPU), ! CPU test class
;    0231  1
;    0232  1  BYTE (0),
;    0233  1  BYTE (MEN$K_TstCla_Memory), ! Memory test class
;    0234  1
;    0235  1  BYTE (0),
;    0236  1  BYTE (MEN$K_TstCla_Disk), ! Disk test class
;    0237  1
;    0238  1  BYTE (0),
;    0239  1  BYTE (MEN$K_TstCla_Tape), ! Tape test class
;    0240  1
;    0241  1  BYTE (0),
;    0242  1  BYTE (MEN$K_TstCla_Comm), ! Communication test class
;    0243  1
;    0244  1  BYTE (0),
;    0245  1  BYTE (MEN$K_TstCla_Real), ! Realtime test class
;    0246  1
;    0247  1  BYTE (0),
;    0248  1  BYTE (MEN$K_TstCla_Term), ! Terminal test class
;    0249  1
;    0250  1  BYTE (0),
;    0251  1  BYTE (MEN$K_TstCla_Printer), ! Printer test class
;    0252  1
;    0253  1  BYTE (0),
;    0254  1  BYTE (MEN$K_TstCla_Card), ! Card Reader test class
;    0255  1
;    0256  1  BYTE (0),
;    0257  1  BYTE (MEN$K_TstCla_IO_Adaptor), ! I/O Adaptor test class
;    0258  1
;    0259  1  BYTE (0),
;    0260  1  BYTE (MEN$K TstCla_All)  ! All test classes
;    0261  1  );
```

```
;    0262  1 %SBTTL 'DEVICE TEST PREPARATION Table '
;    0263  1
;    0264  1 !+
;    0265  1 ! The device test preparation table contains the codes and respective
;    0266  1 ! preparation text pointers describing the required hardware test preparation
;    0267  1 ! required before test is initiated on the hardwre.
;    0268  1 ! Because the CRD MENU code is loadable it is necessary to invoke the
;    0269  1 ! Initialization routine associated with this table following loading
;    0270  1 ! and before the table is accessed (see MENTSTINI module).
;    0271  1 !-
;    0272  1
;    0273  1 GLOBAL
;    0274  1     MEN$GA_PrepTbl : $MEN_PrepTbl_ATTR
;    0275  1         INITIAL (
;    0276  1
;    0277  1  BYTE (MEN$K_DevTstPrep_Null), ! Device Test Prep Code
;    0278  1  LONG (MEN$GT_DevTstPrep_Null), ! Device Test Prep Text
;    0279  1
;    0280  1  BYTE (MEN$K_DevTstPrep_1), ! Device Test Prep Code
;    0281  1  LONG (MEN$GT_DevTstPrep_1), ! Device Test Prep Text
;    0282  1
;    0283  1  BYTE (MEN$K_DevTstPrep_2), ! Device Test Prep Code
;    0284  1  LONG (MEN$GT_DevTstPrep_2), ! Device Test Prep Text
;    0285  1
;    0286  1  BYTE (MEN$K_DevTstPrep_3), ! Device Test Prep Code
;    0287  1  LONG (MEN$GT_DevTstPrep_3), ! Device Test Prep Text
;    0288  1
;    0289  1  BYTE (MEN$K_DevTstPrep_4), ! Device Test Prep Code
;    0290  1  LONG (MEN$GT_DevTstPrep_4), ! Device Test Prep Text
;    0291  1
;    0292  1  BYTE (MEN$K_DevTstPrep_5), ! Device Test Prep Code
;    0293  1  LONG (MEN$GT_DevTstPrep_5), ! Device Test Prep Text
;    0294  1
;    0295  1  BYTE (MEN$K_DevTstPrep_6), ! Device Test Prep Code
;    0296  1  LONG (MEN$GT_DevTstPrep_6), ! Device Test Prep Text
;    0297  1
;    0298  1  BYTE (MEN$K_DevTstPrep_7), ! Device Test Prep Code
;    0299  1  LONG (MEN$GT_DevTstPrep 7) ! Device Test Prep Text
;    0300  1  );
```

```
;    0301  1 %SBTTL 'DEVICE TEST PREPARATION Table Initialization Routine'
;    0302  1
;    0303  1 GLOBAL ROUTINE MEN$PrepTbl_Init =
;    0304  1
;    0305  1 !++
;    0306  1 ! FUNCTIONAL DESCRIPTION:
;    0307  1 !
;    0308  1 !    Since the CRD MENU TEST control code is a loadable image,
;    0309  1 !    link-time resolved address pointers must have an offset added
;    0310  1 !    to them.  This is because the address pointers are resolved
;    0311  1 !    at link time with reference to a zero base address.  However,
;    0312  1 !    at run time the code will be loaded at an address starting at
;    0313  1 !    CRD$AL_CRD_Dispatch_Vectors.  Therefore, for all tables with text
;    0314  1 !    address pointers, add an offset equal to
;    0315  1 !    the value represented by the address location
;    0316  1 !    CRD$AL_CRD_Dispatch_Vectors!
;    0317  1 !
;    0318  1 ! INPUT PARAMETERS
;    0319  1 !
;    0320  1 ! OUTPUT PARAMETERS
;    0321  1 !
;    0322  1 ! IMPLICIT INPUTS
;    0323  1 !
;    0324  1 ! EXPLICIT OUTPUTS
;    0325  1 !
;    0326  1 ! SIDE EFFECTS
;    0327  1 !
;    0328  1 !    PrepTbl_Init is specified True.
;    0329  1 !
;    0330  1 ! COMPLETION CODES
;    0331  1 !
;    0332  1 !    CRD$K_Success
;    0333  1 !--
```

```
;    0334  2 BEGIN
;    0335  2     LOCAL
;    0336  2  Text_Ptr : REF VECTOR [, LONG],
;    0337  2  Index;
;    0338  2
;    0339  2     OWN
;    0340  2         PrepTbl_Init : BYTE INITIAL (BYTE(False));
;    0341  2       ! Flag location indicating whether
;    0342  2       ! Table Text Address pointers
;    0343  2       ! have been initialized with
;    0344  2       ! offset address  values to account
;    0345  2       ! for the fact that the code
;    0346  2       ! is loaded at a dynamic address
;    0347  2       ! location.
;    0348  2       ! 1 or True  = Table Initialized
;    0349  2       ! 0 or False = Table not initialized
;    0350  2
;    0351  2     !+
;    0352  2     ! Perform initialization only once
;    0353  2     !-
;    0354  2
;    0355  3     IF (NOT .PrepTbl_Init)
;    0356  2     THEN
;    0357  3  BEGIN
;    0358  3  !+
;    0359  3  ! Initialize Text Address Pointers for the FUNCTIONAL TEST Menu Table
;    0360  3  !-
;    0361  3
;    0362  3  Index = -1;
;    0363  3
;    0364  3  WHILE ((Index = .Index + 1) LSS (MEN$GK_PrepTbl_Entries)) DO
;    0365  4      BEGIN
;    0366  4      Text_Ptr = MEN$GA_PrepTbl [.Index, PrepTbl$L_Entry_Text];
;    0367  4      ! Address of text pointer
;    0368  4      Text_Ptr [0] = .Text_Ptr [0] + CRD$AL_CRD_Dispatch_Vectors;
;    0369  4      ! Add offset
;    0370  3      END;
;    0371  3  PrepTbl_Init = True;  ! Indicate that Init done
;    0372  2  END;
;    0373  2
;    0374  2     RETURN (CRD$K_Success);
;    0375  1 END;


        .PSECT  WORK,NOEXE,2

        0011F    .BLKB   1
    00  00120 MEN$GA_TMTBL::
        .BYTE   0               ;
    01  00121     .BYTE   1           ;
    00  00122     .BYTE   0           ;
    02  00123     .BYTE   2           ;
    00  00124     .BYTE   0           ;
    03  00125     .BYTE   3           ;
```

```
        00  00126     .BYTE   0           ;
        04  00127     .BYTE   4           ;
        00  00128     .BYTE   0           ;
        05  00129     .BYTE   5           ;
        00  0012A     .BYTE   0           ;
        06  0012B     .BYTE   6           ;
        00  0012C     .BYTE   0           ;
        07  0012D     .BYTE   7           ;
        00  0012E     .BYTE   0           ;
        08  0012F     .BYTE   8           ;
        00  00130     .BYTE   0           ;
        09  00131     .BYTE   9           ;
        00  00132     .BYTE   0           ;
        0A  00133     .BYTE   10          ;
        00  00134     .BYTE   0           ;
        0B  00135     .BYTE   11          ;
            00136   .BLKB   2
        00  00138 MEN$GA_PREPTBL::
            .BYTE   0           ;
         00000000G 00139     .ADDRESS MEN$GT_DEVTSTPREP_NULL     ;
        01  0013D     .BYTE   1           ;
         00000000G 0013E     .ADDRESS MEN$GT_DEVTSTPREP_1        ;
        02  00142     .BYTE   2           ;
         00000000G 00143     .ADDRESS MEN$GT_DEVTSTPREP_2        ;
        03  00147     .BYTE   3           ;
         00000000G 00148     .ADDRESS MEN$GT_DEVTSTPREP_3        ;
        04  0014C     .BYTE   4           ;
         00000000G 0014D     .ADDRESS MEN$GT_DEVTSTPREP_4        ;
        05  00151     .BYTE   5           ;
         00000000G 00152     .ADDRESS MEN$GT_DEVTSTPREP_5        ;
        06  00156     .BYTE   6           ;
         00000000G 00157     .ADDRESS MEN$GT_DEVTSTPREP_6        ;
        07  0015B     .BYTE   7           ;
         00000000G 0015C     .ADDRESS MEN$GT_DEVTSTPREP_7        ;
        00  00160 PREPTBL_INIT:
            .BYTE   0           ;

            .EXTRN  MEN$GT_DEVTSTPREP_NULL
            .EXTRN  MEN$GT_DEVTSTPREP_1
            .EXTRN  MEN$GT_DEVTSTPREP_2
            .EXTRN  MEN$GT_DEVTSTPREP_3
            .EXTRN  MEN$GT_DEVTSTPREP_4
            .EXTRN  MEN$GT_DEVTSTPREP_5
            .EXTRN  MEN$GT_DEVTSTPREP_6
            .EXTRN  MEN$GT_DEVTSTPREP_7

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0004 00000     .ENTRY  MEN$PREPTBL_INIT, Save R2        ; 0303
            29 00000000'  EF  E8 00002     BLBS    PREPTBL_INIT, 3$         ; 0355
            50       01  CE 00009     MNEGL   #1, INDEX           ; 0362
            16  11 0000C     BRB     2$              ; 0364
            51       50       05  C5 0000E 1$:   MULL3   #5, INDEX, R1           ; 0366
            52 00000000'EF41  9E 00012     MOVAB   MEN$GA_PREPTBL+1[R1], TEXT_PTR       ;
            51 00000000G  EF  9E 0001A     MOVAB   CRD$AL_CRD_DISPATCH_VECTORS, R1        ; 0368
```

```
            62      51  C0 00021    ADDL2   R1, (TEXT_PTR)          ;
            50  D6 00024 2$:   INCL    INDEX                   ; 0364
            08      50  D1 00026    CMPL    INDEX, #8           ;
            E3  19 00029    BLSS    1$                      ;
      00000000'  EF      01  90 0002B    MOVB    #1, PREPTBL_INIT        ; 0371
            50      01  D0 00032 3$:   MOVL    #1, R0                  ; 0374
         04 00035    RET                 ; 0375

; Routine Size:  54 bytes,    Routine Base:  CODE + 0036
```

```
;    0376  1 %SBTTL 'FUNCTIONAL TEST MAIN MENU Routine'
;    0377  1
;    0378  1 GLOBAL ROUTINE MEN$FuncTest_Menu =
;    0379  1
;    0380  1 !++
;    0381  1 ! FUNCTIONAL DESCRIPTION:
;    0382  1 !
;    0383  1 !    The Functional Test Main Menu allows the operator to choose various
;    0384  1 !    activites associated with Functional Testing:
;    0385  1 !
;    0386  1 ! Exit CRD MENU TEST
;    0387  1 ! Print identified System Hardware
;    0388  1 ! Print hardware test preparation for supported hardware
;    0389  1 ! Invoke TEST MENU for selecting and testing supported hardware
;    0390  1 ! TEST ALL  hardware
;    0391  1 !
;    0392  1 !    This Routine performs the following:
;    0393  1 !
;    0394  1 ! . Types the Functional Test Main Menu
;    0395  1 ! . Solicit operator input until valid selection is made
;    0396  1 ! . Determine course of action on the basis of valid selection
;    0397  1 !
;    0398  1 ! INPUT PARAMETERS
;    0399  1 !
;    0400  1 ! OUTPUT PARAMETERS
;    0401  1 !
;    0402  1 ! IMPLICIT INPUTS
;    0403  1 !
;    0404  1 ! CRD MENU TEST Initialization must have been performed (see MENTSTINI
;    0405  1 ! module).
;    0406  1 !
;    0407  1 ! EXPLICIT OUTPUTS
;    0408  1 !
;    0409  1 ! If 'Exit' selection is chosen CRD$STOP is called.
;    0410  1 !
;    0411  1 ! If any of the 'print' selections are chosen then the respective
;    0412  1 ! information is printed and Functional Test Menu is repeated.
;    0413  1 !
;    0414  1 ! If TEST MENU or TEST ALL selection is chosen and successfully completed
;    0415  1 ! then the Test Queue Table is available pointed to by MEN$GA_Test_Queue.
;    0416  1 ! The testing header is printed, MEN$GB_Test_In_Progress is
;    0417  1 !  specified True, and return is made to the caller.
;    0418  1 !
;    0419  1 ! SIDE EFFECTS
;    0420  1 !
;    0421  1 !
;    0422  1 ! COMPLETION CODES
;    0423  1 !
;    0424  1 ! CRD$K_FTMRet_Test Start Testing
;    0425  1 !  CRD$K_FTMRet_Failure Fatal Functional Test Menu Failure
;    0426  1 !
;    0427  1 !--
```

```
;    0428  2 BEGIN
;    0429  2 LOCAL
;    0430  2     Entry_Ptr         : REF $MEN_FTMTbl_Entry_ATTR,
;    0431  2     FTM_Table_Index,
;    0432  2     Valid_Status,
;    0433  2     Valid_Alpha_Upper : VECTOR [CH$ALLOCATION (1)],
;    0434  2     Valid_Alpha_Lower : VECTOR [CH$ALLOCATION (1)];
;    0435  2
;    0436  2     !++
;    0437  2     !-------------------------------------------------------------------------
;    0438  2     ! LOCAL ROUTINE:
;    0439  2     ! Type_Main_Menu_Selections:
;    0440  2     !  Types the Main Menu Selections.
;    0441  2     !-------------------------------------------------------------------------
;    0442  2     !--
;    0443  2
;    0444  2     ROUTINE Type_Main_Menu_Selections : NOVALUE =
;    0445  3     BEGIN    ! Routine Type_Main_Menu_Selections
;    0446  3  REGISTER
;    0447  3      FTM_Table_Index,
;    0448  3      Sel_Alpha_Ptr,
;    0449  3      Sel_Text_Ptr;
;    0450  3
;    0451  3  CRD$Disable_Ctrl_C ();  ! Disable Control C's during menu typeout
;    0452  3
;    0453  3  $CRD_Message (New_Line, MEN$GT_FTMTitle);
;    0454  3      ! Print the title
;    0455  3  FTM_Table_Index = -1;  ! Initialize index
;    0456  3
;    0457  3  WHILE ((FTM_Table_Index = .FTM_Table_Index + 1) LSS (MEN$GK_FTMTbl_Entries)) DO
;    0458  4      BEGIN
;    0459  4      Sel_Alpha_Ptr = MEN$GA_FTMTbl [.FTM_Table_Index, FTMTbl$B_Entry_Alpha];
;    0460  4      Sel_Text_Ptr  = .MEN$GA_FTMTbl [.FTM_Table_Index, FTMTbl$L_Entry_Text];
;    0461  4
; P  0462  4      $CRD_Message (New_Line,
; P  0463  4      MEN$GT_FTMenu_Selection,
; P  0464  4       1,  ! Length of selection choice Alpha
; P  0465  4      .Sel_Alpha_Ptr,! Address of Alpha
; P  0466  4      .Sel_Text_Ptr ! Address of selection text ASCIC
;    0467  4     );
;    0468  3      END;    ! End of WHILE loop
;    0469  3
;    0470  3  $CRD_Message (New_Line, MEN$GT_FTM_Choice);
;    0471  3      ! Print operator directive
;    0472  3  CRD$Enable_Ctrl_C ();  ! Enable Control C's again
;    0473  2     END;    ! Routine Type_Main_Menu_Selections


            .EXTRN  CRD$DISABLE_CTRL_C
            .EXTRN  MEN$GT_FTMTITLE
            .EXTRN  CRD$GT_NEW_LINE_MESSAGE
            .EXTRN  CRD$PRINT, MEN$GT_FTMENU_SELECTION
            .EXTRN  MEN$GT_FTM_CHOICE
            .EXTRN  CRD$ENABLE_CTRL_C
```

```
                OFFC 00000 TYPE_MAIN_MENU_SELECTIONS:
                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 0444
00000000G       EF  16 00002    JSB     CRD$DISABLE_CTRL_C          ; 0451
00000000G       EF  9F 00008    PUSHAB  CRD$GT_NEW_LINE_MESSAGE      ; 0453
                01  DD 0000E    PUSHL   #1                          ;
                1A  DD 00010    PUSHL   #26                         ;
00000000G       FF      03  FB 00012    CALLS   #3, @CRD$PRINT      ;
00000000G       EF  9F 00019    PUSHAB  MEN$GT_FTMTITLE             ;
                01  DD 0001F    PUSHL   #1                          ;
                1A  DD 00021    PUSHL   #26                         ;
00000000G       FF      03  FB 00023    CALLS   #3, @CRD$PRINT      ;
                52      01  CE 0002A    MNEGL   #1, FTM_TABLE_INDEX ; 0455
                3C  11 0002D    BRB     2$                          ; 0457
                50      52      06  C5 0002F 1$:   MULL3   #6, FTM_TABLE_INDEX, R0     ; 0459
                53 00000000'EF40  9E 00033    MOVAB   MEN$GA_FTMTBL[R0], SEL_ALPHA_PTR  ;
00000000'EF40   9F 0003B    PUSHAB  MEN$GA_FTMTBL+2[R0]             ; 0460
                54      9E  D0 00042    MOVL    @(SP)+, SEL_TEXT_PTR ;
00000000G       EF  9F 00045    PUSHAB  CRD$GT_NEW_LINE_MESSAGE      ; 0467
                01  DD 0004B    PUSHL   #1                          ;
                1A  DD 0004D    PUSHL   #26                         ;
00000000G       FF      03  FB 0004F    CALLS   #3, @CRD$PRINT      ;
                18  BB 00056    PUSHR   #^M<R3,R4>                  ;
                01  DD 00058    PUSHL   #1                          ;
00000000G       EF  9F 0005A    PUSHAB  MEN$GT_FTMENU_SELECTION      ;
                01  DD 00060    PUSHL   #1                          ;
                1A  DD 00062    PUSHL   #26                         ;
00000000G       FF      06  FB 00064    CALLS   #6, @CRD$PRINT      ;
                52  D6 0006B 2$:   INCL    FTM_TABLE_INDEX         ; 0457
                05      52  D1 0006D    CMPL    FTM_TABLE_INDEX, #5 ;
                BD  19 00070    BLSS    1$                          ;
00000000G       EF  9F 00072    PUSHAB  CRD$GT_NEW_LINE_MESSAGE      ; 0470
                01  DD 00078    PUSHL   #1                          ;
                1A  DD 0007A    PUSHL   #26                         ;
00000000G       FF      03  FB 0007C    CALLS   #3, @CRD$PRINT      ;
00000000G       EF  9F 00083    PUSHAB  MEN$GT_FTM_CHOICE           ;
                01  DD 00089    PUSHL   #1                          ;
                1A  DD 0008B    PUSHL   #26                         ;
00000000G       FF      03  FB 0008D    CALLS   #3, @CRD$PRINT      ;
00000000G       EF  16 00094    JSB     CRD$ENABLE_CTRL_C           ; 0472
                04 0009A    RET                 ; 0473

; Routine Size:  155 bytes,    Routine Base:  CODE + 006C
```

```
;    0474  2      !++
;    0475  2      !-----------------------------------------------------------------------
;    0476  2      ! Start of MAIN ROUTINE:
;    0477  2      !-----------------------------------------------------------------------
;    0478  2      !--
;    0479  2
;    0480  2      MEN$GB_FncTst_Init = True;  ! Must set this here so that correct ^C table will be used
;    0481  2
;    0482  2      !+
;    0483  2      ! Big Main Menu selection loop: Keep looping until the user selects a set of
;    0484  2      ! tests to execute. Then exit this procedure. The only other reason to exit
;    0485  2      ! this loop is if the user wishes to exit CRD Menu Test.
;    0486  2      !-
;    0487  2
;    0488  2      DO
;    0489  3  BEGIN
;    0490  3
;    0491  3  !+
;    0492  3  ! Prompt for operator input until a valid selection is made.
;    0493  3  !-
;    0494  3
;    0495  3  Valid_Status = False;
;    0496  3
;    0497  3  DO
;    0498  4      BEGIN
;    0499  4      !+
;    0500  4      ! Print the Menu Selections from the FUNCTIONAL TEST Menu Table.
;    0501  4      !-
;    0502  4
;    0503  4      Type_Main_Menu_Selections ();
;    0504  4
;    0505  4      !+
;    0506  4      ! Now prompt the user for a selection:
;    0507  4      !-
;    0508  4
; P  0509  5      IF ($MEN_ASKSTR (
; P  0510  5    Msg_Adr = MEN$GT_FTMPrompt,
; P  0511  5    Buf_Adr = MEN$GT_OperInput_Buffer,
;    0512  5    Def_Adr = T_Prompt_Default))
;    0513  4      THEN
;    0514  5    BEGIN
;    0515  5    !+
;    0516  5    ! If soft console terminal, clear screen before even looking
;    0517  5    ! at whether the input was valid or not.
;    0518  5    !-
;    0519  5    CRD$Screen_Pause (SCREEN$K_Clear_Screen);
;    0520  5
;    0521  5    !+
;    0522  5    ! Operator input must be a valid Functional Test Menu Table
;    0523  5    ! entry alpha (lower or uppercase)
;    0524  5    !-
;    0525  5
;    0526  5    FTM_Table_Index = -1;
;    0527  6    WHILE (
;    0528  6     ((FTM_Table_Index = .FTM_Table_Index + 1) LSS (MEN$GK_FTMTbl_Entries)) AND
```

```
;    0529  6     (NOT .Valid_Status))
;    0530  5    DO
;    0531  6        BEGIN
;    0532  6        Entry_Ptr = MEN$GA_FTMTbl [.FTM_Table_Index, FTMTbl$B_Entry_Alpha];
;    0533  6        Valid_Alpha_Upper = .Entry_Ptr [FTMTbl$B_Entry_Alpha];
;    0534  6        Valid_Alpha_Lower = .Valid_Alpha_Upper + %X'20';
;    0535  6
;    0536  6        Valid_Status =
;    0537  7     (CH$EQL (1, Valid_Alpha_Upper, .MEN$GT_OperInput_Buffer [0], MEN$GT_OperInput_Buffer [1]) OR
;    0538  6      CH$EQL (1, Valid_Alpha_Lower, .MEN$GT_OperInput_Buffer [0], MEN$GT_OperInput_Buffer [1]));
;    0539  5        END;  ! WHILE loop
;    0540  4    END;    ! IF statement
;    0541  4
;    0542  5       IF (NOT .Valid_Status)
;    0543  4       THEN
;    0544  4    !+
;    0545  4    ! If <CR/LF> or ^C was typed don't type message, just prompt again.
;    0546  4    ! Both of these operator inputs will load the operator buffer
;    0547  4    ! with a default.
;    0548  4    !-
;    0549  4
;    0550  5    IF NOT (CH$EQL (.MEN$GT_OperInput_Buffer [0], MEN$GT_OperInput_Buffer [1],
;    0551  5              .T_Prompt_Default [0],        T_Prompt_Default [1]))
;    0552  4    THEN
;    0553  4        $CRD_Message (New_Line, MEN$GT_InvOperInput);
;    0554  4       END    ! DO statement
;    0555  3   UNTIL (.Valid_Status);  ! Repeat until valid
;    0556  3
;    0557  3      !+
;    0558  3      ! Now use the menu selection table entry chosen to decide what to do
;    0559  3      !-
;    0560  3
;    0561  3   SELECT (.Entry_Ptr [FTMTbl$B_Entry_FTMSel]) OF
;    0562  3       SET
;    0563  3       [MEN$K_FTMSel_Exit]:
;    0564  3    CRD$Stop (MENU$K_Exit_Operator_Stop);
;    0565  3
;    0566  3       [MEN$K_FTMSel_System_Hdwr ]:
;    0567  3        MEN$Print_System_Hdwr ();
;    0568  3
;    0569  3       [MEN$K_FTMSel_Prep]:
;    0570  3        MEN$Print_DevTstPrep ();
;    0571  3
;    0572  3       [MEN$K_FTMSel_Test]:
;    0573  4        BEGIN
;    0574  4    !+
;    0575  4    ! Do the TEST MENU
;    0576  4        !-
;    0577  5        IF NOT (MEN$Test_Menu ())
;    0578  4    THEN
;    0579  4        RETURN (MEN$K_FTMRet_Failure);
;    0580  3    END;
;    0581  3
;    0582  3       [MEN$K_FTMSel FncTst_All]:
;    0583  3    !+
```

```
;    0584  3    ! Build Test Queue for testing all devices
;    0585  3        !-
;    0586  3    MEN$BldTstQ_FncTst_All ();
;    0587  3
;    0588  3        [MEN$K_FTMSel_Test, MEN$K_FTMSel_FncTst_All]:
;    0589  4    BEGIN
;    0590  4    !+
;    0591  4    ! Print preliminary messages before test
;    0592  4    !-
;    0593  4    MEN$Scratch_Media_Msg ();
;    0594  4        ! Type a warning message
;    0595  4        ! for all selected units that
;    0596  4        ! require scratch media
;    0597  4    $CRD_Print (MEN$GT_CtrlC_FncTst);
;    0598  4        ! Tell the operator how to interrupt
;    0599  4    $CRD_Message (New_Line, MEN$GT_FuncTest_Begin);
;    0600  4        ! Type the Func. Test Title
;    0601  4    MEN$Print_Runtime_Total ();
;    0602  4    MEN$GB_Test_In_Progress =True;
;    0603  4        ! Indicate that testing is now in progress
;    0604  5    IF (.CRD$GB_CRD_Debug)
;    0605  4    THEN
;    0606  4        MEN$Dump_TstQ_HST ();
;    0607  4
;    0608  4    RETURN (MEN$K_FTMRet_Test);
;    0609  3    END;
;    0610  3        TES;    ! End of SELECT
;    0611  3
;    0612  3   END     ! Main DO loop
;    0613  2       UNTIL (False);    ! Loop forever
;    0614  2
;    0615  2       RETURN (CRD$K_Failure);
;    0616  1 END;      ! Functional Test Menu Routine



            .EXTRN  MEN$GB_FNCTST_INIT
            .EXTRN  MEN$GT_FTMPROMPT
            .EXTRN  DS$BREAK, DS$ASKSTR
            .EXTRN  CRD$SCREEN_PAUSE
            .EXTRN  MEN$GT_INVOPERINPUT
            .EXTRN  CRD$STOP, MEN$BLDTSTQ_FNCTST_ALL
            .EXTRN  MEN$SCRATCH_MEDIA_MSG
            .EXTRN  MEN$GT_CTRLC_FNCTST
            .EXTRN  MEN$GT_FUNCTEST_BEGIN
            .EXTRN  MEN$PRINT_RUNTIME_TOTAL
            .EXTRN  MEN$GB_TEST_IN_PROGRESS
            .EXTRN  CRD$GB_CRD_DEBUG
            .EXTRN  MEN$DUMP_TSTQ_HST

              01FC 00000     .ENTRY  MEN$FUNCTEST_MENU, Save R2,R3,R4,R5,R6,R7,- ; 0378
             R8
            5E       08  C2 00002     SUBL2   #8, SP          ;
        00000000G    EF      01  90 00005     MOVB    #1, MEN$GB_FNCTST_INIT      ; 0480
            58  D4 0000C 1$:   CLRL    VALID_STATUS            ; 0495
```

```
    FF52   CF       00   FB 0000E 2$:   CALLS   #0, TYPE_MAIN_MENU_SELECTIONS        ; 0503
       00000000G  9F       00   FB 00013    CALLS   #0, @#DS$BREAK                  ; 0512
           7E  D4 0001A    CLRL    -(SP)           ;
       00000000'  EF   9F 0001C    PUSHAB  T_PROMPT_DEFAULT          ;
           7E  D4 00022    CLRL    -(SP)           ;
           7E 48    8F   9A 00024    MOVZBL  #72, -(SP)          ;
       00000000'  EF   9F 00028    PUSHAB  MEN$GT_OPERINPUT_BUFFER         ;
       00000000G  EF   9F 0002E    PUSHAB  MEN$GT_FTMPROMPT          ;
       00000000G  9F       06   FB 00034    CALLS   #6, @#DS$ASKSTR          ;
           52       50   D0 0003B    MOVL    R0, STATUS          ;
       00000000G  9F       00   FB 0003E    CALLS   #0, @#DS$BREAK          ;
           5F       52   E9 00045    BLBC    STATUS, 7$          ;
           01   DD 00048    PUSHL   #1              ; 0519
       00000000G  EF       01   FB 0004A    CALLS   #1, CRD$SCREEN_PAUSE          ;
           54       01   CE 00051    MNEGL   #1, FTM_TABLE_INDEX         ; 0526
           47   11 00054    BRB     6$              ; 0537
           50         54       06   C5 00056 3$:   MULL3   #6, FTM_TABLE_INDEX, R0        ; 0532
           57 00000000'EF40   9E 0005A    MOVAB   MEN$GA_FTMTBL[R0], ENTRY_PTR        ;
           6E       67   9A 00062    MOVZBL  (ENTRY_PTR), VALID_ALPHA_UPPER       ; 0533
     04    AE         6E       20   C1 00065    ADDL3   #32, VALID_ALPHA_UPPER, VALID_ALPHA_LOWER   ; 0534
           50 00000000'   EF   9A 0006A    MOVZBL  MEN$GT_OPERINPUT_BUFFER, R0         ; 0537
           56   D4 00071    CLRL    R6              ;
    50           00         6E       01   2D 00073    CMPC5   #1, VALID_ALPHA_UPPER, #0, R0, -        ;
       00000000'  EF       00078    MEN$GT_OPERINPUT_BUFFER+1         ;
           02   12 0007D    BNEQ    4$              ;
           56   D6 0007F    INCL    R6              ;
           50 00000000'   EF   9A 00081 4$:   MOVZBL  MEN$GT_OPERINPUT_BUFFER, R0         ; 0538
           55   D4 00088    CLRL    R5              ;
    50           00    04    AE       01   2D 0008A    CMPC5   #1, VALID_ ALPHA_LOWER, #0, R0, -        ;
       00000000'  EF       00090    MEN$GT_OPERINPUT_BUFFER+1         ;
           02   12 00095    BNEQ    5$              ;
           55   D6 00097    INCL    R5              ;
           58         55       56   C9 00099 5$:   BISL3   R6, R5, VALID_STATUS         ;
           54   D6 0009D 6$:   INCL    FTM_TABLE_INDEX         ; 0528
           05       54   D1 0009F    CMPL    FTM_TABLE_INDEX, #5          ;
           03   18 000A2    BGEQ    7$              ;
           AF       58   E9 000A4    BLBC    VALID_STATUS, 3$          ; 0529
           46       58   E8 000A7 7$:   BLBS    VALID_STATUS, 9$          ; 0542
           51 00000000'   EF   9A 000AA    MOVZBL  MEN$GT_OPERINPUT_BUFFER, R1         ; 0550
           50 00000000'   EF   9A 000B1    MOVZBL  T_PROMPT_DEFAULT, R0          ; 0551
    50           00 00000000'   EF       51   2D 000B8    CMPC5   R1, MEN$GT_OPERINPUT_BUFFER+1, #0, R0, -    ;
       00000000'  EF       000C1    T_PROMPT_DEFAULT+1          ;
           22   13 000C6    BEQL    8$              ;
       00000000G  EF   9F 000C8    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0553
           01   DD 000CE    PUSHL   #1              ;
           1A   DD 000D0    PUSHL   #26             ;
       00000000G  FF       03   FB 000D2    CALLS   #3, @CRD$PRINT          ;
       00000000G  EF   9F 000D9    PUSHAB  MEN$GT_INVOPERINPUT         ;
           01   DD 000DF    PUSHL   #1              ;
           1A   DD 000E1    PUSHL   #26             ;
       00000000G  FF       03   FB 000E3    CALLS   #3, @CRD$PRINT          ;
           03       58   E8 000EA 8$:   BLBS    VALID_STATUS, 9$          ; 0555
         FF1E  31 000ED    BRW     2$              ;
           52 01    A7   9A 000F0 9$:   MOVZBL  1(ENTRY_PTR), R2          ; 0561
           01       52   91 000F4    CMPB    R2, #1          ; 0563
```

```
                09  12 000F7     BNEQ    10$             ;
                0B  DD 000F9     PUSHL   #11             ; 0564
            00000000G  EF        01  FB 000FB     CALLS   #1, CRD$STOP            ;
                04       52  91 00102 10$:    CMPB    R2, #4              ; 0566
                07  12 00105     BNEQ    11$             ;
            00000000V  EF        00  FB 00107     CALLS   #0, MEN$PRINT_SYSTEM_HDWR        ; 0567
                02       52  91 0010E 11$:    CMPB    R2, #2              ; 0569
                07  12 00111     BNEQ    12$             ;
            00000000V  EF        00  FB 00113     CALLS   #0, MEN$PRINT_DEVTSTPREP         ; 0570
                03       52  91 0011A 12$:    CMPB    R2, #3              ; 0572
                0E  12 0011D     BNEQ    13$             ;
            00000000V  EF        00  FB 0011F     CALLS   #0, MEN$TEST_MENU           ; 0577
                04       50  E8 00126     BLBS    R0, 13$             ;
                50       04  D0 00129     MOVL    #4, R0              ; 0579
             04 0012C     RET                 ;
                05       52  91 0012D 13$:    CMPB    R2, #5              ; 0582
                07  12 00130     BNEQ    14$             ;
            00000000G  EF        00  FB 00132     CALLS   #0, MEN$BLDTSTQ_FNCTST_ALL       ; 0586
                03       52  91 00139 14$:    CMPB    R2, #3              ; 0588
                08  13 0013C     BEQL    15$             ;
                05       52  91 0013E     CMPB    R2, #5              ;
                03  13 00141     BEQL    15$             ;
              FEC6  31 00143     BRW     1$              ;
            00000000G  EF        00  FB 00146 15$:    CALLS   #0, MEN$SCRATCH_MEDIA_MSG        ; 0593
            00000000G  EF  9F 0014D     PUSHAB  MEN$GT_CTRLC_FNCTST         ; 0597
                01  DD 00153     PUSHL   #1              ;
                1A  DD 00155     PUSHL   #26             ;
            00000000G  FF        03  FB 00157     CALLS   #3, @CRD$PRINT          ;
            00000000G  EF  9F 0015E     PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0599
                01  DD 00164     PUSHL   #1              ;
                1A  DD 00166     PUSHL   #26             ;
            00000000G  FF        03  FB 00168     CALLS   #3, @CRD$PRINT          ;
            00000000G  EF  9F 0016F     PUSHAB  MEN$GT_FUNCTEST_BEGIN           ;
                01  DD 00175     PUSHL   #1              ;
                1A  DD 00177     PUSHL   #26             ;
            00000000G  FF        03  FB 00179     CALLS   #3, @CRD$PRINT          ;
            00000000G  EF        00  FB 00180     CALLS   #0, MEN$PRINT_RUNTIME_TOTAL          ; 0601
            00000000G  EF        01  90 00187     MOVB    #1, MEN$GB_TEST_IN_PROGRESS          ; 0602
                07 00000000G  EF  E9 0018E     BLBC    CRD$GB_CRD_DEBUG, 16$           ; 0604
            00000000G  EF        00  FB 00195     CALLS   #0, MEN$DUMP_TSTQ_HST           ; 0606
                50       03  D0 0019C 16$:    MOVL    #3, R0              ; 0608
             04 0019F     RET                 ; 0616

; Routine Size:  416 bytes,    Routine Base:  CODE + 0107
```

```
;    0617  1 %SBTTL 'TEST MENU Routine'
;    0618  1
;    0619  1 GLOBAL ROUTINE MEN$Test_Menu =
;    0620  1
;    0621  1 !++
;    0622  1 ! FUNCTIONAL DESCRIPTION:
;    0623  1 !
;    0624  1 ! The TEST MENU allows the operator to select and start testing of
;    0625  1 ! one or all supported system hardware:
;    0626  1 !
;    0627  1 !    . Determine valid TEST MENU Table entries
;    0628  1 !
;    0629  1 !    . Print the TEST MENU selections
;    0630  1 !
;    0631  1 !    . Solicit input and validate.
;    0632  1 !
;    0633  1 !    . Build Test Queue in preparation for testing
;    0634  1 !
;    0635  1 ! INPUT PARAMETERS
;    0636  1 !
;    0637  1 ! OUTPUT PARAMETERS
;    0638  1 !
;    0639  1 ! IMPLICIT INPUTS
;    0640  1 !
;    0641  1 ! EXPLICIT OUTPUTS
;    0642  1 !
;    0643  1 ! Test Queue Table pointed to by MEN$GA_Test_Queue
;    0644  1 !
;    0645  1 ! SIDE EFFECTS
;    0646  1 !
;    0647  1 ! COMPLETION CODES
;    0648  1 !
;    0649  1 ! CRD$K_Success Test Menu Routine Successful
;    0650  1 ! CRD$K_Fa'lure Test Menu Routine Failure
;    0651  1 !
;    0652  1 ! -
```

```
;    0653  2 BEGIN
;    0654  2    LOCAL
;    0655  2  Device_Number,
;    0656  2  Entry_Ptr : REF $MEN_TMTbl_Entry_ATTR,
;    0657  2  DDB_Ptr  : REF Device_Data_Block_Structure;
;    0658  2
;    0659  2     REGISTER
;    0660  2  Valid_Status;
;    0661  2
;    0662  2     !++
;    0663  2     !-------------------------------------------------------------------- --
;    0664  2     ! LOCAL ROUTINE:
;    0665  2     ! Type_Test_Menu_Selections:
;    0666  2     !  Types the Test Menu Selections.
;    0667  2     !----------------------------------------------------------------------
;    0668  2     !--
;    0669  2
;    0670  2     ROUTINE Type_Test_Menu_Selections : NOVALUE =
;    0671  3  BEGIN    ! Routine Type_Test_Menu_Selections
;    0672  3  REGISTER
;    0673  3   PTable_Ptr : REF $DS_HPO_Decl (),
;    0674  3   SupBlk_Ptr : REF $MEN_SupBlk_Attr;
;    0675  3
;    0676  3  LOCAL
;    0677  3   DDB_Ptr  : REF Device_Data_Block_Structure,
;    0678  3   Entry_Ptr : REF $MEN_TmTbl_Entry_Attr,
;    0679  3   SelAlpha_Index : VECTOR [CH$ALLOCATION (1)],
;    0680  3
;    0681  3   Last_TstCla,
;    0682  3   TmTbl_Index,
;    0683  3   SelAlpha_Ptr,
;    0684  3   SelDevNumb,
;    0685  3   Hdwr_Name_Ptr,
;    0686  3   Dev_Name,
;    0687  3   TstCla_Name_Ptr,
;    0688  3   Screen_Counter;
;    0689  3
;    0690  3  !+
;    0691  3  ! Determine valid TEST MENU Table entries and load a selection Alpha for each.
;    0692  3  ! An entry is valid if at least one DDB is found to have a support block and the
;    0693  3  ! entry's test class code, or if the entry's test class code is 'Test All'.
;    0694  3  !-
;    0695  3
;    0696  3  SelAlpha_Index = (%C'A'); ! Start the selection Alpha with 'A'.
;    0697  3  TMTbl_Index = -1;  ! Initialize the index
;    0698  3
;    0699  3  WHILE ((TMTbl_Index = .TMTbl_Index + 1) LSS MEN$GK_TMTbl_Entries) DO
;    0700  4   BEGIN
;    0701  4   Entry_Ptr = MEN$GA_TMTbl [.TMTbl_Index, TMTbl$B_Entry_Alpha];
;    0702  4
;    0703  4   IF (MEN$FndDDB_Tcl ( %REF(.Entry_Ptr [TMTbl$B_Entry_TstCla]), DDB_Ptr)) OR
;    0704  5      (.Entry_Ptr [TMTbl$B_Entry_TstCla] EQL MEN$K_TstCla_All)
;    0705  4   THEN
;    0706  5    BEGIN
;    0707  5    Entry_Ptr [TMTb $B_Entry_A pha] - .SelA pha_Index;
```

```
;    0708  5     SelAlpha_Index = .SelAlpha_Index + 1;
;    0709  5     END
;    0710  4    ELSE
;    0711  4     Entry_Ptr [TMTbl$B_Entry_Alpha] = 0;
;    0712  3    END;    ! WHILE loop
;    0713  3
;    0714  3    !+
;    0715  3    ! Print the title
;    0716  3    !-
;    0717  3
;    0718  3    CRD$Disable_Ctrl_C ();  ! Disable Control C's during Menu typeout
;    0719  3
;    0720  3    $CRD_Message ( New_Line, MEN$GT_TMTitle);
;    0721  3
;    0722  3    Screen_Counter = 5;  ! Init line number on which output will occur
;    0723  3
;    0724  3    !+
;    0725  3    ! Get a valid table entry and print menu selection text
;    0726  3    !-
;    0727  3
;    0728  3    Last_TstCla = MEN$K_TstCla_Null;
;    0729  3       ! Start with Null Test Class for this
;    0730  3       ! comparison check location
;    0731  3    TMTbl_Index = -1; ! Initialize the index again
;    0732  3
;    0733  3    WHILE ((TMTbl_Index = .TMTbl_Index +1) LSS MEN$GK_TMTbl_Entries) DO
;    0734  4     BEGIN
;    0735  4     Entry_Ptr = MEN$GA_TMTbl [.TMTbl_Index, TMTbl$B_Entry_Alpha];
;    0736  4
;    0737  5     IF (.Entry_Ptr [TMTbl$B_Entry_Alpha] NEQ 0)
;    0738  4     THEN  ! If valid entry then print text
;    0739  5      BEGIN
;    0740  6      IF (.Last_TstCla NEQ .Entry_Ptr [TMTbl$B_Entry_TstCla])
;    0741  5      THEN
;    0742  6       BEGIN
;    0743  6       !+
;    0744  6       ! Before printing more, check for space on the screen
;    0745  6       !-
;    0746  7       IF (.Screen_Counter GEQ CRD$K_Numb_Screen_Lines)
;    0747  6       THEN
;    0748  7        BEGIN
;    0749  7        CRD$Screen_Pause (SCREEN$K_TM_Msg);
;    0750  7        $CRD_Message ( New_Line, MEN$GT_TMTitle);
;    0751  7        Screen_Counter = 5; ! Title takes up 2 lines
;    0752  7        END
;    0753  6       ELSE
;    0754  7        BEGIN
;    0755  7        Screen_Counter = .Screen_Counter + 1;
;    0756  6        END;
;    0757  6       !+
;    0758  6       ! If new test class then leave extra space
;    0759  6       !
;    0760  6       $CRD_Message (New_Line, MEN$GT_Null);
;    0761  6
;    0762  5       END;
```

```
;    0763  5
;    0764  5     !+
;    0765  5     ! Print Test All menu selection text if 'Test All' test class
;    0766  5     !-
;    0767  5
;    0768  6     IF (.Entry_Ptr [TMTbl$B_Entry_TstCla] EQL MEN$K_TstCla_All)
;    0769  5     THEN
;    0770  6      BEGIN
;    0771  6      SelAlpha_Ptr = Entry_Ptr [TMTbl$B_Entry_Alpha];
;    0772  6      SelDevNumb = MEN$K_TMenu_TstAll_DevNumb;
;    0773  6
;    0774  6
;    0775  6      !+
;    0776  6      ! Before printing more, check for space on the screen
;    0777  6      !-
;    0778  7      IF (.Screen_Counter GEQ CRD$K_Numb_Screen_Lines)
;    0779  6      THEN
;    0780  7       BEGIN
;    0781  7       CRD$Screen_Pause (SCREEN$K_TM_Msg);
;    0782  7       $CRD_Message ( New_Line, MEN$GT_TMTitle);
;    0783  7       Screen_Counter = 5; ! Title takes up 2 lines
;    0784  7       END
;    0785  6      ELSE
;    0786  7       BEGIN
;    0787  7       Screen_Counter = .Screen_Counter + 1;
;    0788  6       END;
;    0789  6
; P  0790  6      $CRD_Message (New_Line,
; P  0791  6        MEN$GT_TMenu_TstAll,
; P  0792  6        1,  ! Alpha size
; P  0793  6        .SelAlpha_Ptr, ! Alpha address
; P  0794  6        .SelDevNumb  ! Select Device Number
;    0795  6             );
;    0796  6      END ! IF statement
;    0797  5     ELSE
;    0798  6      BEGIN
;    0799  6      !+
;    0800  6      ! If not 'Test All' then:
;    0801  6      ! Search the DDB Database for all supported devices of the test class and
;    0802  6      ! if device is found, print menu selection text.
;    0803  6      !-
;    0804  6
;    0805  6      DDB_Ptr = MEN$GA_DDB_Head;
;    0806  6      WHILE ((DDB_Ptr = .DDB_Ptr [CRD$K_DDB_Flink]) NEQ MEN$GA_DDB_Head) DO
;    0807  6       IF ((SupBlk_Ptr = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr]) NEQ 0) THEN
;    0808  7        IF (CH$EQL (1, Entry_Ptr [TMTbl$B_Entry_TstCla],
;    0809  7             1, SupBlk_Ptr [ SupBlk$B_Test_Class]))
;    0810  6        THEN
;    0811  7         BEGIN
;    0812  7             SelAlpha_Ptr  = Entry_Ptr [TMTbl$B_Entry_Alpha];
;    0813  7             SelDevNumb    = .DDB_Ptr [CRD$K_DDB_Menu_Device_Numb];
;    0814  7             Hdwr_Name_Ptr = SupBlk_Ptr [SupBlk$T_Dev'ce Name];
;    0815  7             MEN$Get_TstCla_Name (
;    0816  7           %REF(.Entry_Ptr [TMTb $B_Entry_TstCla]),
;    0817  7           TstCla_Name_Ptr);
```

```
;    0818  7          PTable_Ptr     = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    0819  7          Dev_Name       = CRD$Get_Device_Name (.PTable_Ptr);
;    0820  7
;    0821  7          !+
;    0822  7          ! Before printing more, check for space on the screen
;    0823  7          !-
;    0824  8          IF (.Screen_Counter GEQ CRD$K_Numb_Screen_Lines)
;    0825  7          THEN
;    0826  8           BEGIN
;    0827  8           CRD$Screen_Pause (SCREEN$K_TM_Msg);
;    0828  8           $CRD_Message ( New_Line, MEN$GT_TMTitle);
;    0829  8           Screen_Counter = 5; ! Title takes up 2 lines
;    0830  8           END
;    0831  7          ELSE
;    0832  8           BEGIN
;    0833  8           Screen_Counter = .Screen_Counter + 1;
;    0834  7           END;
;    0835  7
; P  0836  7              $CRD_Message
; P  0837  7           (
; P  0838  7           New_Line,
; P  0839  7           MEN$GT_TMenu_Selection,
; P  0840  7           1,     ! Alpha size
; P  0841  7           .SelAlpha_Ptr,     ! Alpha address
; P  0842  7           .SelDevNumb,      ! Select Device Number
; P  0843  7           .Hdwr_Name_Ptr,    ! Hardware Name ASCIC Address
; P  0844  7           .TstCla_Name_Ptr, ! Test Class Name ASCIC Address
; P  0845  7           .Dev_Name    ! Device name from Ptable
;    0846  7           );
;    0847  6          END; ! WHILE ... IF ... IF ... BEGIN statements
;    0848  5        END;  ! Not the 'Test All' class IF statement
;    0849  4      END;    ! Valid entry
;    0850  3     END;  ! While-Do Loop
;    0851  3
;    0852  3   $CRD_Message (New_Line, MEN$GT_TM_Choice);
;    0853  3        ! Print operator directive
;    0854  3   CRD$Enable_Ctrl_C ();  ! Enable Control C's aga'n
;    0855  2   END;     ! Routine Type_Test_Menu_Selections



            .EXTRN  MEN$GT_TMTITLE, MEN$GT_NULL
            .EXTRN  MEN$GT_TMENU_TSTALL
            .EXTRN  MEN$GA_DDB_HEAD
            .EXTRN  CRD$GET_DEVICE_NAME
            .EXTRN  MEN$GT_TMENU_SELECTION
            .EXTRN  MEN$GT_TM_CHOICE

              0FFC 00000 TYPE_TEST_MENU_SELECTIONS:
            .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 0670
            5E      14  C2 00002    SUBL2   #20, SP             ;
            52 41   8F  9A 00005    MOVZBL  #65, SELALPHA_INDEX          ; 0696
      04    AE      01  CE 00009    MNEGL   #1, TMTBL_INDEX          ; 0697
            39  11 0000D    BRB     4$              ; 0699
            50 04   AE  D0 0000F 1$:   MOVL    TMTBL_INDEX, R0          ; 0701
```

```
                6E 00000000'EF40  3E 00013    MOVAW   MEN$GA_TMTBL[R0], ENTRY_PTR        ;
            0C  AE  9F 0001B    PUSHAB  DDB_PTR          ; 0703
                50   04  AE       01  C1 0001E    ADDL3   #1, ENTRY_PTR, R0          ;
            0C  AE       60  9A 00023    MOVZBL  (R0), 12(SP)          ;
            0C  AE  9F 00027    PUSHAB  12(SP)          ;
             00000000V  EF       02  FB 0002A    CALLS   #2, MEN$FNDDDB_TCL          ;
                09       50  E8 00031    BLBS    R0, 2$          ;
                50         6E       01  C1 00034    ADDL3   #1, ENTRY_PTR, R0          ; 0704
                0B       60  91 00038    CMPB    (R0), #11          ;
                08  12 0003B    BNEQ    3$          ;
            00  BE       52  90 0003D 2$:   MOVB    SELALPHA_INDEX, @ENTRY_PTR          ; 0707
                52  D6 00041    INCL    SELALPHA_INDEX          ; 0708
                03  11 00043    BRB     4$          ; 0703
            00  BE  94 00045 3$:   CLRB    @ENTRY_PTR          ; 0711
            04  AE  D6 00048 4$:   INCL    TMTBL_INDEX          ; 0699
                0B 04   AE  D1 0004B    CMPL    TMTBL_INDEX, #11          ;
                BE  19 0004F    BLSS    1$          ;
             00000000G  EF  16 00051    JSB     CRD$DISABLE_CTRL_C          ; 0718
             00000000G  EF  9F 00057    PUSHAB  CRD$GT_NEW_LINE_MESSAGE          ; 0720
                01  DD 0005D    PUSHL   #1          ;
                1A  DD 0005F    PUSHL   #26          ;
             00000000G  FF       03  FB 00061    CALLS   #3, @CRD$PRINT          ;
             00000000G  EF  9F 00068    PUSHAB  MEN$GT_TMTITLE          ;
                01  DD 0006E    PUSHL   #1          ;
                1A  DD 00070    PUSHL   #26          ;
             00000000G  FF       03  FB 00072    CALLS   #3, @CRD$PRINT          ;
                56       05  D0 00079    MOVL    #5, SCREEN_COUNTER          ; 0722
                5A  D4 0007C    CLRL    LAST_TSTCLA          ; 0728
            04  AE       01  CE 0007E    MNEGL   #1, TMTBL_INDEX          ; 0731
               01BC  31 00082    BRW     19$          ; 0733
                50 04   AE  D0 00085 5$:   MOVL    TMTBL_INDEX, R0          ; 0735
                6E 00000000'EF40  3E 00089    MOVAW   MEN$GA_TMTBL[R0], ENTRY_PTR        ;
            00  BE  95 00091    TSTB    @ENTRY_PTR          ; 0737
                03  12 00094    BNEQ    6$          ;
               01A8  31 00096    BRW     19$          ;
                50         6E       01  C1 00099 6$:   ADDL3   #1, ENTRY_PTR, R0          ; 0740
        5A           60         08       00  ED 0009D    CMPZV   #0, #8, (R0), LAST_TSTCLA          ;
                59  13 000A2    BEQL    9$          ;
                16       56  D1 000A4    CMPL    SCREEN_COUNTER, #22          ; 0746
                30  19 000A7    BLSS    7$          ;
                05  DD 000A9    PUSHL   #5          ; 0749
             00000000G  EF       01  FB 000AB    CALLS   #1, CRD$SCREEN_PAUSE          ;
             00000000G  EF  9F 000B2    PUSHAB  CRD$GT_NEW_LINE_MESSAGE          ; 0750
                01  DD 000B8    PUSHL   #1          ;
                1A  DD 000BA    PUSHL   #26          ;
             00000000G  FF       03  FB 000BC    CALLS   #3, @CRD$PRINT          ;
             00000000G  EF  9F 000C3    PUSHAB  MEN$GT_TMTITLE          ;
                01  DD 000C9    PUSHL   #1          ;
                1A  DD 000CB    PUSHL   #26          ;
             00000000G  FF       03  FB 000CD    CALLS   #3, @CRD$PRINT          ;
                56       05  D0 000D4    MOVL    #5, SCREEN_COUNTER          ; 0751
                02  11 000D7    BRB     8$          ; 0746
                56  D6 000D9 7$:   INCL    SCREEN_COUNTER          ; 0755
             00000000G  EF  9F 000DB 8$:   PUSHAB  CRD$GT_NEW_LINE_MESSAGE          ; 0760
                01  DD 000E1    PUSHL   #1          ;
```

```
                1A  DD 000E3     PUSHL   #26             ;
          00000000G  FF       03  FB 000E5    CALLS   #3, @CRD$PRINT          ;
          00000000G  EF  9F 000EC    PUSHAB  MEN$GT_NULL             ;
                01  DD 000F2     PUSHL   #1              ;
                1A  DD 000F4     PUSHL   #26             ;
          00000000G  FF       03  FB 000F6    CALLS   #3, @CRD$PRINT          ;
                50        6E      01  C1 000FD 9$:   ADDL3   #1, ENTRY_PTR, R0          ; 0768
                0B       60  91 00101    CMPB    (R0), #11               ;
                68  12 00104    BNEQ    12$             ;
                58       6E  D0 00106    MOVL    ENTRY_PTR, SELALPHA_PTR         ; 0771
                57       01  D0 00109    MOVL    #1, SELDEVNUMB          ; 0772
                16       56  D1 0010C    CMPL    SCREEN_COUNTER, #22             ; 0778
                30  19 0010F    BLSS    10$             ;
                05  DD 00111    PUSHL   #5              ; 0781
          00000000G  EF       01  FB 00113    CALLS   #1, CRD$SCREEN_PAUSE            ;
          00000000G  EF  9F 0011A    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0782
                01  DD 00120     PUSHL   #1              ;
                1A  DD 00122     PUSHL   #26             ;
          00000000G  FF       03  FB 00124    CALLS   #3, @CRD$PRINT          ;
          00000000G  EF  9F 0012B    PUSHAB  MEN$GT_TMTITLE          ;
                01  DD 00131     PUSHL   #1              ;
                1A  DD 00133     PUSHL   #26             ;
          00000000G  FF       03  FB 00135    CALLS   #3, @CRD$PRINT          ;
                56       05  D0 0013C    MOVL    #5, SCREEN_COUNTER          ; 0783
                02  11 0013F    BRB     11$             ; 0778
                56  D6 00141 10$:   INCL    SCREEN_COUNTER          ; 0787
          00000000G  EF  9F 00143 11$:   PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0795
                01  DD 00149     PUSHL   #1              ;
                1A  DD 0014B     PUSHL   #26             ;
           00000000G  FF       03  FB 0014D    CALLS   #3, @CRD$PRINT          ;
                57  DD 00154     PUSHL   SELDEVNUMB              ;
                58  DD 00156     PUSHL   SELALPHA_PTR            ;
                01  DD 00158     PUSHL   #1              ;
           00000000G  EF  9F 0015A    PUSHAB  MEN$GT_TMENU_TSTALL             ;
                01  DD 00160     PUSHL   #1              ;
                1A  DD 00162     PUSHL   #26             ;
           00000000G  FF       06  FB 00164    CALLS   #6, @CRD$PRINT          ;
             00D3  31 0016B     BRW     19$             ; 0768
          0C    AE 00000000G  EF  9E 0016E 12$:   MOVAB   MEN$GA_DDB_HEAD, DDB_PTR        ; 0805
                55 0C    AE  D0 00176    MOVL    DDB_PTR, R5             ; 0806
             00B1  31 0017A     BRW     18$             ;
                55 0C    AE  D0 0017D 13$:   MOVL    DDB_PTR, R5             ; 0807
                53 20    A5  D0 00181    MOVL    32(R5), SUPBLK_PTR              ;
                03  12 00185     BNEQ    14$             ;
             00A4  31 00187     BRW     18$             ;
                50        6E      01  C1 0018A 14$:   ADDL3   #1, ENTRY_PTR, R0          ; 0809
          11    A3       60  91 0018E    CMPB    (R0), 17(SUPBLK_PTR)            ;
                03  13 00192     BEQL    15$             ;
             0097  31 00194     BRW     18$             ;
                58       6E  D0 00197 15$:   MOVL    ENTRY_PTR, SELALPHA_PTR         ; 0812
                57 1C    A5  D0 0019A    MOVL    28(R5), SELDEVNUMB          ; 0813
                54 04    A3  9E 0019E    MOVAB   4(R3), HDWR_NAME_PTR            ; 0814
          10    AE  9F 001A2    PUSHAB  TSTCLA_NAME_PTR         ; 0815
                50   04   AE      01  C1 001A5    ADDL3   #1, ENTRY_PTR, R0          ; 0816
          0C    AE       60  9A 001AA    MOVZBL  (R0), 12(SP)            ;
```

```
      0C   AE  9F 001AE     PUSHAB  12(SP)                ;
       00000000V  EF       02  FB 001B1     CALLS   #2, MEN$GET_TSTCLA_NAME        ;
           52 08   A5  D0 001B8     MOVL    8(R5), PTABLE_PTR          ; 0818
           52  DD 001BC     PUSHL   PTABLE_PTR              ; 0819
       00000000G  EF       01  FB 001BE     CALLS   #1, CRD$GET_DEVICE_NAME        ;
           59       50  D0 001C5     MOVL    R0, DEV_NAME            ;
           16       56  D1 001C8     CMPL    SCREEN_COUNTER, #22          ; 0824
           30  19 001CB     BLSS    16$                ;
           05  DD 001CD     PUSHL   #5               ; 0827
       00000000G  EF       01  FB 001CF     CALLS   #1, CRD$SCREEN_PAUSE          ;
       00000000G  EF  9F 001D6     PUSHAB  CRD$GT_NEW_LINE_MESSAGE          ; 0828
           01  DD 001DC     PUSHL   #1               ;
           1A  DD 001DE     PUSHL   #26              ;
       00000000G  FF       03  FB 001E0     CALLS   #3, @CRD$PRINT          ;
       00000000G  EF  9F 001E7     PUSHAB  MEN$GT_TMTITLE          ;
           01  DD 001ED     PUSHL   #1               ;
           1A  DD 001EF     PUSHL   #26              ;
       00000000G  FF       03  FB 001F1     CALLS   #3, @CRD$PRINT          ;
           56       05  D0 001F8     MOVL    #5, SCREEN_COUNTER          ; 0829
           02  11 001FB     BRB     17$                ; 0824
           56  D6 001FD 16$:   INCL    SCREEN_COUNTER          ; 0833
       00000000G  EF  9F 001FF 17$:   PUSHAB  CRD$GT_NEW_LINE_MESSAGE          ; 0846
           01  DD 00205     PUSHL   #1               ;
           1A  DD 00207     PUSHL   #26              ;
       00000000G  FF       03  FB 00209     CALLS   #3, @CRD$PRINT          ;
           59  DD 00210     PUSHL   DEV_NAME            ;
      14   AE  DD 00212     PUSHL   TSTCLA_NAME_PTR          ;
           54  DD 00215     PUSHL   HDWR_NAME_PTR           ;
           57  DD 00217     PUSHL   SELDEVNUMB            ;
           58  DD 00219     PUSHL   SELALPHA_PTR            ;
           01  DD 0021B     PUSHL   #1               ;
       00000000G  EF  9F 0021D     PUSHAB  MEN$GT_TMENU_SELECTION          ;
           01  DD 00223     PUSHL   #1               ;
           1A  DD 00225     PUSHL   #26              ;
       00000000G  FF       09  FB 00227     CALLS   #9, @CRD$PRINT          ;
      0C   AE       65  D0 0022E 18$:   MOVL    (R5), DDB_PTR          ; 0806
           50 00000000G  EF  9E 00232     MOVAB   MEN$GA_DDB_HEAD, R0          ;
           50       65  D1 00239     CMPL    (R5), R0            ;
           03  13 0023C     BEQL    19$                ;
         FF3C  31 0023E     BRW     13$                ;
      04   AE  D6 00241 19$:   INCL    TMTBL_INDEX          ; 0733
           0B 04   AE  D1 00244     CMPL    TMTBL_INDEX, #11          ;
           03  18 00248     BGEQ    20$                ;
         FE38  31 0024A     BRW     5$                 ;
       00000000G  EF  9F 0024D 20$:   PUSHAB  CRD$GT_NEW_LINE_MESSAGE          ; 0852
           01  DD 00253     PUSHL   #1               ;
           1A  DD 00255     PUSHL   #26              ;
       00000000G  FF       03  FB 00257     CALLS   #3, @CRD$PRINT          ;
       00000000G  EF  9F 0025E     PUSHAB  MEN$GT_TM_CHOICE          ;
           01  DD 00264     PUSHL   #1               ;
           1A  DD 00266     PUSHL   #26              ;
       00000000G  FF       03  FB 00268     CALLS   #3, @CRD$PRINT          ;
       00000000G  EF  16 0026F     JSB     CRD$ENABLE_CTRL_C          ; 0854
        04 00275     RET                ; 0855
```

```
; Routine Size:  630 bytes,    Routine Base:  CODE + 02A7
```

```
;    0856  2      !++
;    0857  2      !--------------------------------------------------------------------
;    0858  2      ! LOCAL ROUTINE:
;    0859  2      !
;    0860  2      ! TMenu_OpInp_Chk: Validate the test menu operator input.
;    0861  2      !
;    0862  2      ! INPUT PARAMETERS:
;    0863  2      !
;    0864  2      !  I_Buffer: Address of ASCIC
;    0865  2      !
;    0866  2      ! OUTPUT PARAMETERS:
;    0867  2      !
;    0868  2      ! I_Entry: Address to return address pointer of selected
;    0869  2      !   entry in TEST MENU Table
;    0870  2      ! I_Number: Address of longword to return decimal device number
;    0871  2      !   specified by operator
;    0872  2      !
;    0873  2      ! COMPLETION CODES:
;    0874  2      !
;    0875  2      ! CRD$K_Success: Operator input valid
;    0876  2      ! CRD$K_Failure: Operator input invalid
;    0877  2      !--------------------------------------------------------------------
;    0878  2
```

```
;    0879  2
;    0880  2     ROUTINE TMenu_OpInp_Chk (I_Buffer, I_Entry, I_Number) =
;    0881  3  BEGIN
;    0882  3  LITERAL
;    0883  3   Numb_Buf_Sz = 20;
;    0884  3
;    0885  3  LOCAL
;    0886  3   Input_End,
;    0887  3   Numb_Buf : VECTOR [Numb_Buf_Sz, BYTE],
;    0888  3   Entry_Ptr : REF $MEN_TMTbl_Entry_ATTR;
;    0889  3
;    0890  3  REGISTER
;    0891  3   Buf_Ptr,
;    0892  3   Input_Alpha,
;    0893  3   Move_Length,
;    0894  3   Numb_Buf_Ptr,
;    0895  3   Numb_Sz,
;    0896  3   Valid_Alpha_Status,
;    0897  3   Valid_Alpha_Upper,
;    0898  3   Valid_Alpha_Lower,
;    0899  3   TMTbl_Index;
;    0900  3
;    0901  3  BIND
;    0902  3   Buffer = .I_Buffer : $MEN_OperInput_Buffer_ATTR,
;    0903  3   Entry  = .I_Entry,
;    0904  3   Number = .I_Number;
;    0905  3
;    0906  3  !+
;    0907  3  ! Separate the Alpha and Numeric operator input fields
;    0908  3  !-
;    0909  3
;    0910  4  IF (.Buffer [0] NEQ 0)
;    0911  3  THEN
;    0912  4   BEGIN
;    0913  4   Input_End     = Buffer [1] + .Buffer [0];
;    0914  4   Buf_Ptr       = Buffer [1];
;    0915  4   Input_Alpha   = CH$RCHAR_A (Buf_Ptr);
;    0916  4   Move_Length   = .Input_End -.Buf_Ptr;
;    0917  4
;    0918  4   Move_Length   = MINU (.Move_Length, Numb_Buf_Sz);
;    0919  4
;    0920  4   Numb_Buf_Ptr = CH$MOVE (.Move_Length, .Buf_Ptr, Numb_Buf);
;    0921  4   Numb_Sz       = .Numb_Buf_Ptr - Numb_Buf;
;    0922  4   END
;    0923  3  ELSE
;    0924  3   RETURN (CRD$K_Failure);
;    0925  3
;    0926  3  !+
;    0927  3  ! Now check that the Alpha character field
;    0928  3  ! represents an entry in the TEST MENU Table.
;    0929  3  !-
;    0930  3
;    0931  3  TMTbl Index = -1;
;    0932  3  Valid Alpha_Status = False;
;    0933  3
```

```
;    0934  3  WHILE (((TMTbl_Index = .TMTbl_Index +1) LSS MEN$GK_TMTbl_Entries) AND (NOT .Valid_Alpha_Status)) DO
;    0935  4   BEGIN
;    0936  4   Entry_Ptr    = MEN$GA_TMTbl [.TMTbl_Index, TMTbl$B_Entry_Alpha];
;    0937  4   Valid_Alpha_Upper = .MEN$GA_TMTbl [.TMTbl_Index, TMTbl$B_Entry_Alpha];
;    0938  4   Valid_Alpha_Lower = .Valid_alpha_Upper + %X'20';
;    0939  4
;    0940  4   IF CH$EQL ( 1, Valid_Alpha_Upper, 1, Input_Alpha) OR
;    0941  4      CH$EQL ( 1, Valid_Alpha_Lower, 1, Input_Alpha)
;    0942  4   THEN
;    0943  5    BEGIN
;    0944  5    Entry = .Entry_Ptr;
;    0945  5    Valid_Alpha_Status = True;
;    0946  4    END;
;    0947  3   END;   ! WHILE ... DO ...
;    0948  3
;    0949  4  IF (NOT .Valid_Alpha_Status)
;    0950  3  THEN
;    0951  3   RETURN (CRD$K_Failure); ! Return the table entry
;    0952  3
;    0953  3   !+
;    0954  3   ! Now check that each ASCII character in the numeric field is a numeric
;    0955  3   ! between 0 and 9.
;    0956  3   !-
;    0957  3
;    0958  3  Numb_Buf_Ptr = Numb_Buf - 1;
;    0959  3
;    0960  3  WHILE ((Numb_Buf_Ptr = CH$PLUS (.Numb_Buf_Ptr, 1)) LSS (Numb_Buf + .Numb_Sz)) DO
;    0961  4   BEGIN
;    0962  4   IF (CH$RCHAR (.Numb_Buf_Ptr) LSS %C'0') OR
;    0963  5      (CH$RCHAR (.Numb_Buf_Ptr) GTR %C'9')
;    0964  4   THEN
;    0965  4    RETURN (CRD$K_Failure);
;    0966  3   END;
;    0967  3
;    0968  3   !+
;    0969  3   ! Convert the ASCII representation of the number to a decimal
;    0970  3   ! and return the decimal device number
;    0971  3   !-
;    0972  3
;    0973  3  MEN$Scan_Numeric_ASC_D (.Numb_Sz, Numb_Buf ,Number);
;    0974  3
;    0975  3  RETURN (CRD$K_Success);
;    0976  2     END;      Routine TMenu_OpInp_Chk


        .EXTRN  MEN$SCAN_NUMERIC_ASC_D

          00FC 00000 TMENU_OPINP_CHK:
        .WORD   Save R2,R3,R4,R5,R6,R7          ; 0880
        5E      14  C2 00002    SUBL2   #20, SP         ;
  04    BC  95 00005    TSTB    aI_BUFFER           ; 0910
        03  12 00008    BNEQ    1$          ;
     009A  31 0000A    BRW     9$          ;
        51 04   BC  9A 0000D 1$:   MOVZBL  aI_BUFFER, R1          ; 0913
```

```
                51 04   AC  C0 00011     ADDL2   I_BUFFER, R1           ;
                51  D6 00015     INCL     INPUT_END           ;
                50   04   AC       01  C1 00017     ADDL3   #1, I_BUFFER, BUF_PTR          ; 0914
                56       80  9A 0001C     MOVZBL  (BUF_PTR)+, INPUT_ALPHA          ; 0915
                51       50  C2 0001F     SUBL2   BUF_PTR, MOVE_LENGTH          ; 0916
                52       51  D0 00022     MOVL    MOVE_LENGTH, R2          ; 0918
                14       52  D1 00025     CMPL    R2, #20           ;
                03  1B 00028     BLEQU   2$              ;
                52       14  D0 0002A     MOVL    #20, R2           ;
                51       52  D0 0002D 2$:    MOVL    R2, MOVE_LENGTH          ;
                6E         60       51  28 00030     MOVC3   MOVE_LENGTH, (BUF_PTR), NUMB_BUF        ; 0920
                50       53  D0 00034     MOVL    R3, NUMB_BUF_PTR          ;
                51       6E  9E 00037     MOVAB   NUMB_BUF, R1          ; 0921
                51         50       51  C3 0003A     SUBL3   R1, NUMB_BUF_PTR, NUMB_SZ         ;
                55       01  CE 0003E     MNEGL   #1, TMTBL_INDEX          ; 0931
                52  D4 00041     CLRL    VALID_ALPHA_STATUS          ; 0932
                27  11 00043     BRB     5$              ; 0940
                57 00000000'EF45  3E 00045 3$:    MOVAW   MEN$GA_TMTBL[TMTBL_INDEX], ENTRY_PTR        ; 0936
     00000000'EF45  3F 0004D     PUSHAW  MEN$GA_TMTBL[TMTBL_INDEX]         ; 0937
                53       9E  9A 00054     MOVZBL  @(SP)+, VALID_ALPHA_UPPER          ;
                54 20    A3  9E 00057     MOVAB   32(R3), VALID_ALPHA_LOWER          ; 0938
                56       53  91 0005B     CMPB    VALID_ALPHA_UPPER, INPUT_ALPHA        ; 0940
                05  13 0005E     BEQL    4$              ;
                56       54  91 00060     CMPB    VALID_ALPHA_LOWER, INPUT_ALPHA        ; 0941
                07  12 00063     BNEQ    5$              ;
          08    BC       57  D0 00065 4$:    MOVL    ENTRY_PTR, @I_ENTRY          ; 0944
                52       01  D0 00069     MOVL    #1, VALID_ALPHA_STATUS         ; 0945
                55  D6 0006C 5$:    INCL    TMTBL_INDEX          ; 0934
                0B       55  D1 0006E     CMPL    TMTBL_INDEX, #11          ;
                03  18 00071     BGEQ    6$              ;
                CF       52  E9 00073     BLBC    VALID_ALPHA_STATUS, 3$          ;
                2E       52  E9 00076 6$:    BLBC    VALID_ALPHA_STATUS, 9$          ; 0949
                50 FF    AE  9E 00079     MOVAB   NUMB_BUF-1, NUMB_BUF_PTR         ; 0958
                52         51       5E  C1 0007D     ADDL3   SP, NUMB_SZ, R2          ; 0960
                0A  11 00081     BRB     8$              ;
                30       60  91 00083 7$:    CMPB    (NUMB_BUF_PTR), #48          ; 0962
                1F  1F 00086     BLSSU   9$              ;
                39       60  91 00088     CMPB    (NUMB_BUF_PTR), #57          ; 0963
                1A  1A 0008B     BGTRU   9$              ;
                50  D6 0008D 8$:    INCL    NUMB_BUF_PTR          ; 0960
                52       50  D1 0008F     CMPL    NUMB_BUF_PTR, R2          ;
                EF  19 00092     BLSS    7$              ;
          0C    AC  DD 00094     PUSHL   I_NUMBER          ; 0973
          04    AE  9F 00097     PUSHAB  NUMB_BUF          ;
                51  DD 0009A     PUSHL   NUMB_SZ         ;
           00000000G  EF       03  FB 0009C     CALLS   #3, MEN$SCAN_NUMERIC_ASC_D        ;
                50       01  D0 000A3     MOVL    #1, R0          ; 0975
            04 000A6     RET             ;
                50  D4 000A7 9$:    CLRL    R0          ; 0976
            04 000A9     RET             ;

; Routine Size:  170 bytes,    Routine Base:  CODE + 051D
```

```
;    0977  2
;    0978  2      !++
;    0979  2      !-------------------------------------------------------------------------
;    0980  2      ! MAIN ROUTINE:
;    0981  2      ! MEN$Test_Menu
;    0982  2      !     Print the Test Menu, get a response from the operator, and validate
;    0983  2      !     the response.
;    0984  2      !-------------------------------------------------------------------------
;    0985  2      !--
;    0986  2
;    0987  2      !+
;    0988  2      ! Get operator input until valid menu selection is made
;    0989  2      !-
;    0990  2
;    0991  2      Valid_Status = False;
;    0992  2
;    0993  2      DO
;    0994  3    BEGIN
;    0995  3    Type_Test_Menu_Selections ();
;    0996  3
; P  0997  4    IF ($MEN_ASKSTR (Msg_Adr = MEN$GT_TMPrompt,
; P  0998  4       Buf_Adr = MEN$GT_OperInput_Buffer,
;    0999  4       Def_Adr = T_Prompt_Default))
;    1000  3    THEN
;    1001  4     BEGIN
;    1002  4     !+
;    1003  4     ! Soft console - before even validating input, clear the screen
;    1004  4     !-
;    1005  4     CRD$Screen_Pause (SCREEN$K_Clear_Screen);
;    1006  4
;    1007  4     !+
;    1008  4     ! Validate the operator input
;    1009  4     !-
;    1010  4
;    1011  5     IF (TMenu_OpInp_Chk (MEN$GT_OperInput_Buffer, Entry_Ptr, Device_Number))
;    1012  4     THEN
;    1013  4      SELECTONE (.Entry_Ptr [TMTbl$B_Entry_TstCla]) OF
;    1014  4       SET
;    1015  4       [MEN$K_TstCla_All]:
;    1016  4        !+
;    1017  4        ! If the 'Test All' TEST MENU entry was chosen check that valid
;    1018  4        ! device number was specified
;    1019  4        !-
;    1020  4
;    1021  4        Valid Status - (.Device_Number EQL MEN$K_TMenu_TstAll_DevNumb);
;    1022  4
;    1023  4       [OTHERWISE] :
;    1024  4        !+
;    1025  4        ! Since the number represents a device verify that
;    1026  4        ! the device is in the DDB database
;    1027  4        !-
;    1028  4
;    1029  4        Valid_Status = MEN$FndDDB_Tcl_SDN
;    1030  4         (xREF (.Entry_Ptr [TMTbl$B_Entry_TstCla]),
;    1031  4          Device_Number,
```

```
;    1032  4          DDB_Ptr);
;    1033  4       TES;
;    1034  3     END;    ! IF statement
;    1035  3
;    1036  4     IF (NOT .Valid_Status)
;    1037  3     THEN
;    1038  3      !+
;    1039  3      ! If <CR/LF> or ^C was typed don't type message, just prompt again.
;    1040  3      ! Both of these operator inputs will load the operator buffer
;    1041  3      ! with the default.
;    1042  3      !-
;    1043  3
;    1044  5      IF (NOT (CH$EQL (.MEN$GT_OperInput_Buffer [0], MEN$GT_OperInput_Buffer [1],
;    1045  4                .T_Prompt_Default [0],        T_Prompt_Default [1])))
;    1046  3      THEN
;    1047  3       $CRD_Message (New_Line, MEN$GT_InvOperInput);
;    1048  3     END    ! DO loop
;    1049  2        UNTIL (.Valid_Status);
;    1050  2
;    1051  2        !+
;    1052  2        ! Now use the menu selection table entry chosen to decide what to do.
;    1053  2        !-
;    1054  2
;    1055  2        SELECTONE (.Entry_Ptr [TMTbl$B_Entry_TstCla]) OF
;    1056  2     SET
;    1057  2     [MEN$K_TstCla_All]:
;    1058  2      !+
;    1059  2      ! Build test queue for all supported hardware
;    1060  2      !-
;    1061  2      MEN$BldTstQ_FncTst_All ();
;    1062  2
;    1063  2     [OTHERWISE]:
;    1064  2      !+
;    1065  2      ! Build test queue for the one hardware device chosen
;    1066  2      !-
;    1067  2      MEN$BldTstQ_FncTst_Dev (.DDB_Ptr);
;    1068  2     TES;
;    1069  2
;    1070  2        RETURN (CRD$K_Success);
;    1071  1 END;     ! Routine MEN$Test_Menu



            .EXTRN  MEN$GT_TMPROMPT
            .EXTRN  MEN$BLDTSTQ_FNCTST_DEV

              001C 00000     .ENTRY  MEN$TEST_MENU, Save R2,R3,R4        ; 0619
            5E       10  C2 00002     SUBL2   #16, SP                 ;
            54  D4 00005     CLRL    VALID_STATUS                    ; 0991
        FCD4    CF       00  FB 00007 1$:   CALLS   #0, TYPE_TEST_MENU_SELECTIONS        ; 0995
           00000000G  9F       00  FB 0000C     CALLS   #0, @#DS$BREAK          ; 0999
            7E  D4 00013     CLRL    -(SP)                           ;
           00000000'  EF  9F 00015     PUSHAB  T_PROMPT_DEFAULT        ;
            7E  D4 0001B     CLRL    -(SP)                           ;
            7E 48   8F  9A 0001D     MOVZBL  #72, -(SP)              ;
```

```
                00000000'  EF  9F 00021    PUSHAB  MEN$GT_OPERINPUT_BUFFER        ;
                00000000G  EF  9F 00027    PUSHAB  MEN$GT_TMPROMPT          ;
                00000000G  9F      06  FB 0002D    CALLS   #6, @#DS$ASKSTR          ;
                    52      50  D0 00034    MOVL    R0, STATUS           ;
                00000000G  9F      00  FB 00037    CALLS   #0, @#DS$BREAK           ;
                    4E      52  E9 0003E    BLBC    STATUS, 4$           ;
                    01  DD 00041    PUSHL   #1              ; 1005
                00000000G  EF      01  FB 00043    CALLS   #1, CRD$SCREEN_PAUSE         ;
               0C   AE  9F 0004A    PUSHAB  DEVICE_NUMBER           ; 1011
               08   AE  9F 0004D    PUSHAB  ENTRY_PTR           ;
                00000000'  EF  9F 00050    PUSHAB  MEN$GT_OPERINPUT_BUFFER        ;
              FEFB   CF      03  FB 00056    CALLS   #3, TMENU_OPINP_CHK          ;
                    31      50  E9 0005B    BLBC    R0, 4$           ;
                    50 04   AE  D0 0005E    MOVL    ENTRY_PTR, R0           ; 1013
                    50  D6 00062    INCL    R0              ;
                    51      60  9A 00064    MOVZBL  (R0), R1            ;
                    0B      51  91 00067    CMPB    R1, #11             ; 1015
                    0C  12 0006A    BNEQ    2$              ;
                    50  D4 0006C    CLRL    R0              ; 1021
                    01 0C   AE  D1 0006E    CMPL    DEVICE_NUMBER, #1           ;
                    18  12 00072    BNEQ    3$              ;
                    50  D6 00074    INCL    R0              ;
                    14  11 00076    BRB     3$              ;
               08   AE  9F 00078 2$:   PUSHAB  DDB_PTR           ; 1030
               10   AE  9F 0007B    PUSHAB  DEVICE_NUMBER           ;
               08   AE      51  D0 0007E    MOVL    R1, 8(SP)           ;
               08   AE  9F 00082    PUSHAB  8(SP)           ;
                00000000V  EF      03  FB 00085    CALLS   #3, MEN$FNDDDB_TCL_SDN       ;
                    54      50  D0 0008C 3$:   MOVL    R0, VALID_STATUS         ;
                    46      54  E8 0008F 4$:   BLBS    VALID_STATUS, 6$         ; 1036
                    51 00000000'  EF  9A 00092    MOVZBL  MEN$GT_OPERINPUT_BUFFER, R1       ; 1044
                    50 00000000'  EF  9A 00099    MOVZBL  T_PROMPT_DEFAULT, R0          ; 1045
            50          00 00000000'  EF      51  2D 000A0    CMPC5   R1, MEN$GT_OPERINPUT_BUFFER+1, #0, R0, -    :
                00000000'  EF      000A9    T_PROMPT_DEFAULT+1          ;
                    22  13 000AE    BEQL    5$              ;
                00000000G  EF  9F 000B0    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 1047
                    01  DD 000B6    PUSHL   #1              ;
                    1A  DD 000B8    PUSHL   #26             ;
                00000000G  FF      03  FB 000BA    CALLS   #3, @#CRD$PRINT          ;
                00000000G  EF  9F 000C1    PUSHAB  MEN$GT_INVOPERINPUT          ;
                    01  DD 000C7    PUSHL   #1              ;
                    1A  DD 000C9    PUSHL   #26             ;
                00000000G  FF      03  FB 000CB    CALLS   #3, @#CRD$PRINT          ;
                    03      54  E8 000D2 5$:   BLBS    VALID_STATUS, 6$         ; 1049
                  FF2F  31 000D5    BRW     1$              ;
                    50 04   AE  D0 000D8 6$:   MOVL    ENTRY_PTR, R0           ; 1055
                    50  D6 000DC    INCL    R0              ;
                    50      60  9A 000DE    MOVZBL  (R0), R0            ;
                    0B      50  91 000E1    CMPB    R0, #11             ; 1057
                    09  12 000E4    BNEQ    7$              ;
                00000000G  EF      00  FB 000E6    CALLS   #0, MEN$BLDTSTQ_FNCTST_ALL       ; 1061
                    0A  11 000ED    BRB     8$              ;
               08   AE  DD 000EF 7$:   PUSHL   DDB_PTR            ; 1067
                00000000G  EF      01  FB 000F2    CALLS   #1, MEN$BLDTSTQ_FNCTST_DEV       ;
                    50      01  D0 000F9 8$:   MOVL    #1, R0              ; 1070
```

```
        04 000FC    RET                 ; 1071

; Routine Size:  253 bytes,    Routine Base:  CODE + 05C7
```

```
;    1072  1 %SBTTL 'Get Test Class ASCII name Routine'
;    1073  1
;    1074  1 GLOBAL ROUTINE MEN$Get_TstCla_Name ( R_CLASS_CODE, I_NAME) =
;    1075  1
;    1076  1 !++
;    1077  1 ! FUNCTIONAL DESCRIPTION:
;    1078  1 !
;    1079  1 !    Returns Test Class ASCII text name for specified Test Class Code.
;    1080  1 !
;    1081  1 ! INPUT PARAMETERS
;    1082  1 !
;    1083  1 !  R_CLASS_CODE Address of byte containing Test Class code
;    1084  1 !
;    1085  1 ! OUTPUT PARAMETERS
;    1086  1 !
;    1087  1 ! I_NAME  Address to return ASCIC Address of test class name
;    1088  1 ! IMPLICIT INPUTS
;    1089  1 !
;    1090  1 ! EXPLICIT OUTPUTS
;    1091  1 !
;    1092  1 ! If the specified test class code is not found an ASCIC null string is returned
;    1093  1 !
;    1094  1 ! SIDE EFFECTS
;    1095  1 !
;    1096  1 ! COMPLETION CODES
;    1097  1 !
;    1098  1 ! CRD$K_Success
;    1099  1 !
;    1100  1 !--
```

```
;    1101  2 BEGIN
;    1102  2     BIND
;    1103  2  Class_Code = .R_CLASS_CODE : BYTE,
;    1104  2  Name       = .I_NAME;
;    1105  2
;    1106  2     SELECTONE .CLASS_CODE OF
;    1107  2  SET
;    1108  2  [MEN$K_TstCla_CPU] :  ! CPU test class
;    1109  2      Name = MEN$GT_TstCla_CPU;
;    1110  2
;    1111  2  [MEN$K_TstCla_Memory] : ! Memory test class
;    1112  2      Name = MEN$GT_TstCla_Memory;
;    1113  2
;    1114  2  [MEN$K_TstCla_Disk] : ! Disk test class
;    1115  2      Name = MEN$GT_TstCla_Disk;
;    1116  2
;    1117  2  [MEN$K_TstCla_Tape] : ! Tape test class
;    1118  2      Name = MEN$GT_TstCla_Tape;
;    1119  2
;    1120  2  [MEN$K_TstCla_Comm] : ! Communication test class
;    1121  2      Name = MEN$GT_TstCla_Comm;
;    1122  2
;    1123  2  [MEN$K_TstCla_Real] : ! Realtime test class
;    1124  2      Name = MEN$GT_TstCla_Real;
;    1125  2
;    1126  2  [MEN$K_TstCla_Term] : ! Terminal test class
;    1127  2      Name = MEN$GT_TstCla_Term;
;    1128  2
;    1129  2  [MEN$K_TstCla_Printer] : ! Printer test class
;    1130  2      Name = MEN$GT_TstCla_Printer;
;    1131  2
;    1132  2  [MEN$K_TstCla_Card] : ! Card Reader test class
;    1133  2      Name = MEN$GT_TstCla_Card;
;    1134  2
;    1135  2  [MEN$K_TstCla_IO_Adaptor] :! I/O Adaptor test class
;    1136  2      Name = MEN$GT_TstCla_IOADAP;
;    1137  2
;    1138  2  [MEN$K_TstCla_All] :  ! All test classes
;    1139  2      Name = MEN$GT_TstCla_All;
;    1140  2
;    1141  2  [OTHERWISE] :
;    1142  2      Name = MEN$GT_TstCla_NullName;
;    1143  2  TES;
;    1144  2
;    1145  2     Return (CRD$K_Success);
;    1146  1     END;


        .EXTRN  MEN$GT_TSTCLA_CPU
        .EXTRN  MEN$GT_TSTCLA_MEMORY
        .EXTRN  MEN$GT_TSTCLA_DISK
        .EXTRN  MEN$GT_TSTCLA_TAPE
        .EXTRN  MEN$GT_TSTCLA_COMM
        .EXTRN  MEN$GT_TSTCLA_REAL
```

```
            .EXTRN  MEN$GT_TSTCLA_TERM
            .EXTRN  MEN$GT_TSTCLA_PRINTER
            .EXTRN  MEN$GT_TSTCLA_CARD
            .EXTRN  MEN$GT_TSTCLA_IOADAP
            .EXTRN  MEN$GT_TSTCLA_ALL
            .EXTRN  MEN$GT_TSTCLA_NULLNAME

              0000 00000     .ENTRY  MEN$GET_TSTCLA_NAME, Save nothing       ; 1074
           50 08    AC  D0 00002     MOVL    I_NAME, R0             ; 1104
           51 04    BC  9A 00006     MOVZBL  @R_CLASS_CODE, R1          ; 1106
           01        51  91 0000A     CMPB    R1, #1                 ; 1108
           0A  12 0000D     BNEQ    1$                      ;
           60 00000000G  EF  9E 0000F     MOVAB   MEN$GT_TSTCLA_CPU, (R0)        ; 1109
         0094  31 00016     BRW     12$                     ;
           02        51  91 00019 1$:   CMPB    R1, #2                 ; 1111
           0A  12 0001C     BNEQ    2$                      ;
           60 00000000G  EF  9E 0001E     MOVAB   MEN$GT_TSTCLA_MEMORY, (R0)        ; 1112
         0085  31 00025     BRW     12$                     ;
           03        51  91 00028 2$:   CMPB    R1, #3                 ; 1114
           09  12 0002B     BNEQ    3$                      ;
           60 00000000G  EF  9E 0002D     MOVAB   MEN$GT_TSTCLA_DISK, (R0)        ; 1115
           77  11 00034     BRB     12$                     ;
           04        51  91 00036 3$:   CMPB    R1, #4                 ; 1117
           09  12 00039     BNEQ    4$                      ;
           60 00000000G  EF  9E 0003B     MOVAB   MEN$GT_TSTCLA_TAPE, (R0)        ; 1118
           69  11 00042     BRB     12$                     ;
           05        51  91 00044 4$:   CMPB    R1, #5                 ; 1120
           09  12 00047     BNEQ    5$                      ;
           60 00000000G  EF  9E 00049     MOVAB   MEN$GT_TSTCLA_COMM, (R0)        ; 1121
           5B  11 00050     BRB     12$                     ;
           06        51  91 00052 5$:   CMPB    R1, #6                 ; 1123
           09  12 00055     BNEQ    6$                      ;
           60 00000000G  EF  9E 00057     MOVAB   MEN$GT_TSTCLA_REAL, (R0)        ; 1124
           4D  11 0005E     BRB     12$                     ;
           07        51  91 00060 6$:   CMPB    R1, #7                 ; 1126
           09  12 00063     BNEQ    7$                      ;
           60 00000000G  EF  9E 00065     MOVAB   MEN$GT_TSTCLA_TERM, (R0)        ; 1127
           3F  11 0006C     BRB     12$                     ;
           08        51  91 0006E 7$:   CMPB    R1, #8                 ; 1129
           09  12 00071     BNEQ    8$                      ;
           60 00000000G  EF  9E 00073     MOVAB   MEN$GT_TSTCLA_PRINTER, (R0)        ; 1130
           31  11 0007A     BRB     12$                     ;
           09        51  91 0007C 8$:   CMPB    R1, #9                 ; 1132
           09  12 0007F     BNEQ    9$                      ;
           60 00000000G  EF  9E 00081     MOVAB   MEN$GT_TSTCLA_CARD, (R0)        ; 1133
           23  11 00088     BRB     12$                     ;
           0A        51  91 0008A 9$:   CMPB    R1, #10                ; 1135
           09  12 0008D     BNEQ    10$                     ;
           60 00000000G  EF  9E 0008F     MOVAB   MEN$GT_TSTCLA_IOADAP, (R0)        ; 1136
           15  11 00096     BRB     12$                     ;
           0B        51  91 00098 10$:  CMPB    R1, #11                ; 1138
           09  12 0009B     BNEQ    11$                     ;
           60 00000000G  EF  9E 0009D     MOVAB   MEN$GT_TSTCLA_ALL, (R0)        ; 1139
           07  11 000A4     BRB     12$                     ;
           60 00000000G  EF  9E 000A6 11$:   MOVAB   MEN$GT_TSTCLA_NULLNAME, (R0)        ; 1142
```

```
            50      01  D0 000AD 12$:   MOVL    #1, R0          ; 1145
          04 000B0    RET                 ; 1146

; Routine Size:  177 bytes,    Routine Base:  CODE + 06C4
```

```
;    1147  1 %SBTTL 'Find DDB for Test Class and Select Device No. Routine'
;    1148  1
;    1149  1 GLOBAL ROUTINE MEN$FndDDB_Tcl_SDN ( R_CLASS_CODE, R_SDN, I_DDB)  =
;    1150  1
;    1151  1 !++
;    1152  1 ! FUNCTIONAL DESCRIPTION:
;    1153  1 !
;    1154  1 !    Searches for DDB which has a support block and the Test Class
;    1155  1 !    code and Select Device Number specified.
;    1156  1 !
;    1157  1 ! INPUT PARAMETERS
;    1158  1 !
;    1159  1 !    R_CLASS_CODE  Address of byte containing Test Class Code.
;    1160  1 !
;    1161  1 !    R_SDN  Address of longword containing  Select Device Number
;    1162  1 !
;    1163  1 ! OUTPUT PARAMETERS
;    1164  1 !
;    1165  1 !    I_DDB  Address of longword to return DDB pointer
;    1166  1 !
;    1167  1 ! IMPLICIT INPUTS
;    1168  1 !
;    1169  1 ! EXPLICIT OUTPUTS
;    1170  1 !
;    1171  1 ! SIDE EFFECTS
;    1172  1 !
;    1173  1 ! COMPLETION CODES
;    1174  1 !
;    1175  1 ! CRD$K_Success DDB found
;    1176  1 ! CRD$K_Failure DDB not found
;    1177  1 !
;    1178  1 !--
```

```
;    1179  1
;    1180  2 BEGIN
;    1181  2     REGISTER
;    1182  2  DDB_Found,
;    1183  2  Queue_BlkPtr  : REF $MEN_Queue_Link_ATTR,
;    1184  2  DDB_Ptr  : REF Device_Data_Block_Structure,
;    1185  2  SupBlk_Ptr  : REF $MEN_SupBlk_Attr;
;    1186  2
;    1187  2     BIND
;    1188  2  Class_Code = .R_Class_Code : BYTE,
;    1189  2  SDN      = .R_SDN,
;    1190  2  DDB      = .I_DDB;
;    1191  2
;    1192  2     DDB_Found    = False;
;    1193  2     Queue_BlkPtr = MEN$GA_DDB_Head;
;    1194  2
;    1195  2     WHILE (((Queue_BlkPtr = .Queue_BlkPtr [Queue$L_Flink]) NEQ MEN$GA_DDB_Head) AND (NOT .DDB_Found)) DO
;    1196  3  BEGIN
;    1197  3  DDB_Ptr = .Queue_BlkPtr;
;    1198  3
;    1199  3  IF ( SupBlk_Ptr = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr]) NEQ 0
;    1200  3  THEN
;    1201  3      DDB_Found =
;    1202  3   (CH$EQL (1, Class_Code, 1, SupBlk_Ptr [SupBlk$B_Test_Class])) AND
;    1203  3   (.SDN EQL .DDB_Ptr [CRD$K DDB Menu_Device_Numb]);
;    1204  2  END;
;    1205  2
;    1206  2     IF .DDB_Found
;    1207  2     THEN
;    1208  2  DDB = .DDB_Ptr
;    1209  2     ELSE
;    1210  2  DDB = 0;
;    1211  2
;    1212  2     RETURN (.DDB_Found);
;    1213  1 END;      ! Routine MEN$FndDDB_Tcl_Sdn



          003C 00000     .ENTRY  MEN$FNDDDB_TCL_SDN, Save R2,R3,R4,R5        ; 1149
        50  D4 00002     CLRL    DDB_FOUND                  ; 1192
        51 00000000G  EF  9E 00004     MOVAB   MEN$GA_DDB_HEAD, QUEUE_BLKPTR        ; 1193
        26  11 0000B     BRB     4$                 ; 1195
        52       51  D0 0000D 1$:   MOVL    QUEUE_BLKPTR, DDB_PTR          ; 1197
        53 20    A2  D0 00010     MOVL    32(DDB_PTR), SUPBLK_PTR        ; 1199
        1D  13 00014     BEQL    4$                 ;
        55  D4 00016     CLRL    R5                 ; 1202
    11  A3 04    BC  91 00018     CMPB    aR_CLASS_CODE, 17(SUPBLK_PTR)          ;
        02  12 0001D     BNEQ    2$                 ;
        55  D6 0001F     INCL    R5                 ;
        54  D4 00021 2$:   CLRL    R4                 ; 1203
    1C  A2 08    BC  D1 00023     CMPL    aR SDN, 28(DDB_PTR)          ;
        02  12 00028     BNEQ    3$                 ;
        54  D6 0002A     INCL    R4                 ;
```

```
            50      55  D2 0002C 3$:   MCOML   R5, DDB_FOUND           ;
            50        54      50  CB 0002F    BICL3   DDB_FOUND, R4, DDB_FOUND       ;
            51      61  D0 00033 4$:   MOVL    (QUEUE_BLKPTR), QUEUE_BLKPTR        ; 1195
            54 00000000G  EF  9E 00036    MOVAB   MEN$GA_DDB_HEAD, R4         ;
            54      51  D1 0003D    CMPL    QUEUE_BLKPTR, R4          ;
            03  13 00040    BEQL    5$              ;
            C8      50  E9 00042    BLBC    DDB_FOUND, 1$           ;
            05      50  E9 00045 5$:   BLBC    DDB_FOUND, 6$             ; 1206
      0C    BC      52  D0 00048    MOVL    DDB_PTR, aI_DDB           ; 1208
        04 0004C    RET             ;
      0C    BC  D4 0004D 6$:   CLRL    aI_DDB            ; 1210
        04 00050    RET             ; 1213

; Routine Size:  81 bytes,    Routine Base:  CODE + 0775
```

```
;    1214  1 %SBTTL 'Find DDB for Test Class Routine'
;    1215  1
;    1216  1 GLOBAL ROUTINE MEN$FndDDB_Tcl ( R_CLASS_CODE,I_DDB)  =
;    1217  1
;    1218  1 !++
;    1219  1 ! FUNCTIONAL DESCRIPTION:
;    1220  1 !
;    1221  1 !    Searches for a DDB which has a support block and the Test Class
;    1222  1 !    code specified. Returns the DDB pointer of
;    1223  1 !    the first DDB found.
;    1224  1 !
;    1225  1 ! INPUT PARAMETERS
;    1226  1 !
;    1227  1 !    R_CLASS_CODE  Address of byte containing Test Class Code.
;    1228  1 !
;    1229  1 !
;    1230  1 ! OUTPUT PARAMETERS
;    1231  1 !
;    1232  1 !    I_DDB  Address of longword to return DDB pointer
;    1233  1 !
;    1234  1 ! IMPLICIT INPUTS
;    1235  1 !
;    1236  1 ! EXPLICIT OUTPUTS
;    1237  1 !
;    1238  1 ! SIDE EFFECTS
;    1239  1 !
;    1240  1 ! COMPLETION CODES
;    1241  1 !
;    1242  1 ! CRD$K_Success DDB found
;    1243  1 ! CRD$K_Failure DDB not found
;    1244  1 !
;    1245  1 !--
```

```
;    1246  2 BEGIN
;    1247  2     REGISTER
;    1248  2  DDB_Found,
;    1249  2  Queue_BlkPtr  : REF $MEN_Queue_Link_ATTR,
;    1250  2  DDB_Ptr  : REF Device_Data_Block_Structure,
;    1251  2  SupBlk_Ptr  : REF $MEN_SupBlk_Attr;
;    1252  2
;    1253  2     BIND
;    1254  2  Class_Code = .R_Class_Code : BYTE,
;    1255  2  DDB    = .I_DDB;
;    1256  2
;    1257  2     DDB_Found = False;
;    1258  2     Queue_BlkPtr = MEN$GA_DDB_Head;
;    1259  2
;    1260  2     WHILE (((Queue_BlkPtr = .Queue_BlkPtr [Queue$L_Flink]) NEQ MEN$GA_DDB_Head) AND (NOT .DDB_Found)) DO
;    1261  3  BEGIN
;    1262  3  DDB_Ptr = .Queue_BlkPtr;
;    1263  3
;    1264  3  IF (SupBlk_Ptr = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr]) NEQ 0
;    1265  3  THEN
;    1266  3      DDB_Found = CH$EQL (1, Class_Code, 1, SupBlk_Ptr [SupBlk$B_Test_Class]);
;    1267  2  END;
;    1268  2
;    1269  2     IF .DDB_Found
;    1270  2     THEN
;    1271  2  DDB = .DDB_Ptr
;    1272  2     ELSE
;    1273  2  DDB = 0;
;    1274  2     RETURN (.DDB_Found);
;    1275  1 END;     ! Routine MEN$FndDDB_Tcl



          001C 00000     .ENTRY  MEN$FNDDDB_TCL, Save R2,R3,R4        ; 1216
       50  D4 00002     CLRL    DDB_FOUND            ; 1257
       51 00000000G  EF  9E 00004     MOVAB   MEN$GA_DDB_HEAD, QUEUE_BLKPTR        ; 1258
       17  11 0000B     BRB     3$              ; 1260
       52       51  D0 0000D 1$:   MOVL    QUEUE_BLKPTR, DDB_PTR          ; 1262
       53 20    A2  D0 00010     MOVL    32(DDB_PTR), SUPBLK_PTR        ; 1264
       0E  13 00014     BEQL    3$              ;
       54  D4 00016     CLRL    R4              ; 1266
    11 A3 04    BC  91 00018     CMPB    aR_CLASS_CODE, 17(SUPBLK_PTR)        ;
       02  12 0001D     BNEQ    2$              ;
       54  D6 0001F     INCL    R4              ;
       50       54  D0 00021 2$:   MOVL    R4, DDB_FOUND            ;
       51       61  D0 00024 3$:   MOVL    (QUEUE_BLKPTR), QUEUE_BLKPTR         ; 1260
       54 00000000G  EF  9E 00027     MOVAB   MEN$GA_DDB_HEAD, R4          ;
       54       51  D1 0002E     CMPL    QUEUE_BLKPTR, R4          ;
       03  13 00031     BEQL    4$              ;
       D7       50  E9 00033     BLBC    DDB_FOUND, 1$          ;
       05       50  E9 00036 4$:   BLBC    DDB_FOUND, 5$          ; 1269
    08 BC       52  D0 00039     MOVL    DDB_PTR, aI_DDB          ; 1271
      04 0003D     RET                 ;
```

```
      08   BC  D4 0003E 5$:   CLRL    aI_DDB          ; 1273
        04 00041    RET                ; 1275

; Routine Size:  66 bytes,    Routine Base:  CODE + 07C6
```

```
;    1276  1 %SBTTL 'Print Hardware Device Test Preparation Routine'
;    1277  1
;    1278  1 GLOBAL ROUTINE MEN$Print_DevTstPrep =
;    1279  1
;    1280  1 !++
;    1281  1 ! FUNCTIONAL DESCRIPTION:
;    1282  1 !
;    1283  1 !    Prints the hardware device test preparation required for
;    1284  1 !    each supported device on the system.  This preparation is required
;    1285  1 !    before the testing for the device is initiated.
;    1286  1 !
;    1287  1 !    Format:
;    1288  1 !
;    1289  1 ! Hardware : Hardware Name 1 ,Hardware Name 2...Hardware Name N
;    1290  1 ! Preparaton : 'Text'
;    1291  1 !
;    1292  1 ! Hardware : Hardware Name 1 ,Hardware Name 2...Hardware Name N
;    1293  1 ! Preparaton : 'Text'
;    1294  1 !      !
;    1295  1 !      !
;    1296  1 !      !
;    1297  1 !      !
;    1298  1 ! Hardware : Hardware Name 1 ,Hardware Name 2...Hardware Name N
;    1299  1 ! Preparaton : 'Text'
;    1300  1 !
;    1301  1 !    Each preparation  text represents a unique preparation text printout found in the
;    1302  1 !    Device Test Preparation Table (MEN$GA_PrepTbl).
;    1303  1 !
;    1304  1 ! INPUT PARAMETERS
;    1305  1 !
;    1306  1 !
;    1307  1 ! OUTPUT PARAMETERS
;    1308  1 !
;    1309  1 !
;    1310  1 ! IMPLICIT INPUTS
;    1311  1 !
;    1312  1 !    DDB Database (MEN$GA_DDB_Head).
;    1313  1 !
;    1314  1 !    Device Test Preparation Table (MEN$GA_PrepTbl).
;    1315  1 !
;    1316  1 ! EXPLICIT OUTPUTS
;    1317  1 !
;    1318  1 !    Terminal printout.
;    1319  1 !
;    1320  1 ! SIDE EFFECTS
;    1321  1 !
;    1322  1 ! COMPLETION CODES
;    1323  1 !
;    1324  1 ! CRD$K_Success
;    1325  1 !
;    1326  1 !--
```

```
;    1327  2 BEGIN
;    1328  2     LITERAL
;    1329  2  Pattern_Buf_Sz = 24;
;    1330  2
;    1331  2     LOCAL
;    1332  2  Screen_Counter,
;    1333  2  PrepTbl_Index,
;    1334  2  DevTstPrep_Code,
;    1335  2  HdwrNam_Found,
;    1336  2  HdwrNam_String_Buf_Ptr ,
;    1337  2  HdwrNam_String_Buf_End,
;    1338  2  HdwrNam_String_Sz,
;    1339  2  Pattern_String_Sz,
;    1340  2  Pattern_Buf_Ptr,
;    1341  2
;    1342  2  Pattern_Buf    : VECTOR [Pattern_Buf_Sz, BYTE];
;    1343  2
;    1344  2     REGISTER
;    1345  2  HdwrNam_Ptr    : REF VECTOR [,BYTE],
;    1346  2  Entry_Ptr    : REF $MEN_PrepTbl_Entry_ATTR,
;    1347  2  DDB_Ptr    : REF Device_Data_Block_Structure,
;    1348  2  PTable_Ptr    : REF $DS_HPO_DECL (),
;    1349  2  SupBlk_Ptr    : REF $MEN_SupBlk_Attr,
;    1350  2  TstCnfg_Ptr    : REF $MEN_TstCnfg_ATTR;
;    1351  2
;    1352  2     Screen_Counter = 1;   ! Init soft console line number on
;    1353  2      ! which output will occur.
;    1354  2     !+
;    1355  2     ! Get a device test preparation code from the table
;    1356  2     !-
;    1357  2
;    1358  2     PrepTbl_Index = -1;
;    1359  2
;    1360  2     WHILE ((PrepTbl_Index = .PrepTbl_Index + 1) LSS (MEN$GK_PrepTbl_Entries) ) DO
;    1361  3  BEGIN
;    1362  3  Entry_Ptr = MEN$GA_PrepTbl [.PrepTbl_Index, PrepTbl$B_Entry_Code];
;    1363  3    DevTstPrep_Code = .Entry_Ptr [PrepTbl$B_Entry_Code];
;    1364  3  HdwrNam_String_Buf_Ptr = HdwrNam_String_Buf;
;    1365  3      ! Point to buffer
;    1366  3  HdwrNam_String_Buf_End = HdwrNam_String_Buf + HdwrNam_String_Buf_Sz;
;    1367  3      ! Calculate end of buffer
;    1368  3
;    1369  3  !+
;    1370  3  ! Get all unique Hardware Names from the DDB's which
;    1371  3  ! have a support block and the device test preparation code
;    1372  3  ! specified.  Build a string of hardware device names.
;    1373  3  ! The hardware names are separated by commas and each name
;    1374  3  ! has a trailing space.
;    1375  3  !-
;    1376  3
;    1377  3  HdwrNam_Found = False;  ! Assume that no hardware name was found
;    1378  3      ! and therefore no string buffer built
;    1379  3  DDB_Ptr = MEN$GA_DDB_Head;
;    1380  3
;    1381  3  WHILE ((DDB_Ptr = .DDB_Ptr [CRD$K_DDB_Flink]) NEQ MEN$GA_DDB_Head) DO
```

```
;    1382  4      BEGIN
;    1383  4      PTable_Ptr = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    1384  4      ! Point to P-Table
;    1385  4      IF ( SupBlk_Ptr = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr]) NEQ 0
;    1386  4      THEN
;    1387  5    BEGIN
;    1388  5    TstCnfg_Ptr = SupBlk_Ptr [SupBlk$B_StrtFrst_CnfgBlk];
;    1389  5
;    1390  5    !+
;    1391  5    ! If the DDB has the specified prep. code and if the
;    1392  5    ! DDB's hardware name is not already located in the
;    1393  5    ! string buffer then load the buffer with the hardware name.
;    1394  5    !-
;    1395  5
;    1396  5    HdwrNam_Ptr = PTable_Ptr [HP$T_Type];
;    1397  5       ! Point to hardware name
;    1398  5    Pattern_Buf_Ptr =
;    1399  5     CH$COPY (.HdwrNam_Ptr [0], HdwrNam_Ptr [1], %C',', (.HdwrNam_Ptr [0] + 1), Pattern_Buf);
;    1400  5       ! Load the hardware name into the pattern
;    1401  5       ! buffer with a trailing comma
;    1402  5    Pattern_String_Sz = .Pattern_Buf_Ptr - Pattern_Buf;
;    1403  5       ! Get pattern size including trailing Comma
;    1404  5
;    1405  5    !+
;    1406  5    ! Check for prep. code and that hardware name not in
;    1407  5    ! string buffer and that there is adequate room in buffer
;    1408  5    !-
;    1409  5
;    1410  5    IF (.DevTstPrep_Code EQL .TstCnfg_Ptr [TstCnfg$B_Prep]) AND
;    1411  6       (CH$FAIL
;    1412  6     (CH$FIND_SUB
;    1413  6      (HdwrNam_String_Buf_Sz,
;    1414  6       HdwrNam_String_Buf,
;    1415  6       (.Pattern_Buf_Ptr - Pattern_Buf),
;    1416  6       Pattern_Buf)
;    1417  6     )
;    1418  5       ) AND
;    1419  6       ((.HdwrNam_String_Buf_End - .HdwrNam_String_Buf_Ptr) GEQ (.HdwrNam_Ptr [0] + 2))
;    1420  5    THEN
;    1421  6        BEGIN
;    1422  6        !+
;    1423  6        ! Load DDB's hardware name from pattern buffer into string buffer.
;    1424  6        ! Load Hardware name into buffer, with trailing comma and space.
;    1425  6        !-
;    1426  6
;    1427  6        HdwrNam_String_Buf_Ptr =
;    1428  6      CH$COPY (.Pattern_String_Sz,
;    1429  6         Pattern_Buf,
;    1430  6         %C' ',
;    1431  6        .Pattern_String_Sz+1,
;    1432  6        .HdwrNam_String_Buf_Ptr);
;    1433  6        HdwrNam_Found = True;
;    1434  6       ! Indicate that at least one hardware name was found
;    1435  5        END;
;    1436  4    END;
```

```
;    1437  3      END;
;    1438  3
;    1439  3  IF .HdwrNam_Found
;    1440  3  THEN
;    1441  4      BEGIN
;    1442  4      HdwrNam_String_Sz = .HdwrNam_String_Buf_Ptr - HdwrNam_String_Buf - 2;
;    1443  4      ! Get String size in bytes ignoring the trailing comma and space
;    1444  4      !+
;    1445  4      ! Before printing anything make sure there is enough room on the screen
;    1446  4      !-
;    1447  4
;    1448  4      ! *** This is temporary kludge until
;    1449  4      ! *** CRD$Screen_Pause (SCREEN$K_Count_Line) gets written.
;    1450  4      SELECTONE  .PrepTbl_Index OF
;    1451  4   SET
;    1452  4   [0]: Screen_Counter = .Screen_Counter + 3;
;    1453  4   [1]: Screen_Counter = .Screen_Counter + 6;
;    1454  4   [2]: Screen_Counter = .Screen_Counter + 4;
;    1455  4   [3]: Screen_Counter = .Screen_Counter + 6;
;    1456  4   [4]: Screen_Counter = .Screen_Counter + 5;
;    1457  4   [5]: Screen_Counter = .Screen_Counter + 5;
;    1458  4   [6]: Screen_Counter = .Screen_Counter + 7;
;    1459  4   TES;
;    1460  4
;    1461  5      IF (.Screen_Counter GEQ CRD$K_Numb_Screen_Lines)
;    1462  4      THEN
;    1463  5   BEGIN
;    1464  5    CRD$Screen_Pause (SCREEN$K_Press_Return);
;    1465  5    Screen_Counter = 1;
;    1466  5      ! Init output line number again
;    1467  4   END;
;    1468  4
;    1469  4
;    1470  4      !+
;    1471  4      ! Print the Hardware Device name string
;    1472  4      !-
;    1473  4
;    1474  4 !****     $CRD_Message (New_Line, MEN$GT_DevTstPrep_Name, .HdwrNam_String_Sz, HdwrNam_String_Buf);
;    1475  4
;    1476  4      $CRD_Print (CRD$GT_New_Line);
;    1477  4      $CRD_Print (MEN$GT_DevTstPrep_Name, .HdwrNam_String_Sz, HdwrNam_String_Buf);
;    1478  4
;    1479  4      !+
;    1480  4      ! Print the Hardware Device Test Preparation Text
;    1481  4      !-
;    1482  4
;    1483  4      $CRD_Print (MEN$GT_DevTstPrep_Text);
;    1484  4      $CRD_Print (.Entry_Ptr [PrepTbl$L_Entry_Text]);
;    1485  3      END;   ! End of IF .HdwrNam_Found THEN BEGIN-END
;    1486  2  END;    ! End of WHILE (PrepTbl_Index...) DO BEGIN-END
;    1487  2
;    1488  2
;    1489  2     $CRD_Print (MEN$GT_End_of_List); ! Print [End of list]
;    1490  2
;    1491  2     CRD$Screen_Pause (SCREEN$K_Press_Return_MM);
```

```
;   1492  2      ! Pause before returning to MAIN MENU.
;   1493  2
;   1494  2      RETURN (CRD$K_Success);
;   1495  1 END;



          .EXTRN  CRD$GT_NEW_LINE
          .EXTRN  MEN$GT_DEVTSTPREP_NAME
          .EXTRN  MEN$GT_DEVTSTPREP_TEXT
          .EXTRN  MEN$GT_END_OF_LIST

            0FFC 00000     .ENTRY  MEN$PRINT_DEVTSTPREP, Save R2,R3,R4,R5,R6,- ; 1278
           R7,R8,R9,R10,R11         ;
          5E       3C  C2 00002     SUBL2   #60, SP          ;
       20 AE       01  D0 00005     MOVL    #1, SCREEN_COUNTER          ; 1352
       1C AE       01  CE 00009     MNEGL   #1, PREPTBL_INDEX           ; 1358
          0171  31 0000D     BRW     15$             ; 1360
          50   1C   AE       05  C5 00010 1$:   MULL3   #5, PREPTBL_INDEX, R0          ; 1362
          57 00000000'EF40  9E 00015     MOVAB   MEN$GA_PREPTBL[R0], ENTRY_PTR        ;
       04 AE       67  9A 0001D     MOVZBL  (ENTRY_PTR), DEVTSTPREP_CODE          ; 1363
       18 AE 00000000'  EF  9E 00021     MOVAB   HDWRNAM_STRING_BUF, HDWRNAM_STRING_BUF_PTR  ; 1364
       08 AE 00000000'  EF  9E 00029     MOVAB   HDWRNAM_STRING_BUF+256, -           ; 1366
           HDWRNAM_STRING_BUF_END       ;
       10 AE  D4 00031     CLRL    HDWRNAM_FOUND           ; 1377
          58 00000000G  EF  9E 00034     MOVAB   MEN$GA_DDB_HEAD, DDB_PTR          ; 1379
          76  11 0003B     BRB     4$              ; 1381
          59 08   A8  D0 0003D 2$:   MOVL    8(DDB_PTR), PTABLE_PTR          ; 1383
          5A 20   A8  D0 00041     MOVL    32(DDB_PTR), SUPBLK_PTR         ; 1385
          6C  13 00045     BEQL    4$              ;
          5B 17   AA  9E 00047     MOVAB   23(R10), TSTCNFG_PTR            ; 1388
          56 26   A9  9E 0004B     MOVAB   38(R9), HDWRNAM_PTR           ; 1396
          51       66  9A 0004F     MOVZBL  (HDWRNAM_PTR), R1           ; 1399
          50       66  9A 00052     MOVZBL  (HDWRNAM_PTR), R0           ;
          50  D6 00055     INCL    R0              ;
    50          2C   01   A6      51  2C 00057    MOVC5   R1, 1(HDWRNAM_PTR), #44, R0, PATTERN_BUF    ;
      24 AE      0005D             ;
          6E       53  D0 0005F     MOVL    R3, PATTERN_BUF_PTR           ;
          50 24   AE  9E 00062     MOVAB   PATTERN_BUF, R0           ; 1402
          54         6E      50  C3 00066     SUBL3   R0, PATTERN_BUF_PTR, R4         ;
       14 AE       54  D0 0006A     MOVL    R4, PATTERN_STRING_SZ           ;
       04 AE           6B          08      00  ED 0006E     CMPZV   #0, #8, (TSTCNFG_PTR), DEVTSTPREP_CODE       ; 1410
          3D  12 00074     BNEQ    4$              ;
00000000' EF   0100   8F   24   AE       54  39 00076     MATCHC  R4, PATTERN_BUF, #256, HDWRNAM_STRING_BUF    ; 1413
          03  13 00082     BEQL    3$              ;
          53       54  D0 00084     MOVL    R4, R3          ;
          53       54  C2 00087 3$:   SUBL2   R4, R3          ;
          27  12 0008A     BNEQ    4$              ; 1417
          51   08   AE 18   AE  C3 0008C     SUBL3   HDWRNAM_STRING_BUF_PTR, -         ; 1419
           HDWRNAM_STRING_BUF_END, R1       ;
          50       66  9A 00092     MOVZBL  (HDWRNAM_PTR), R0           ;
          50       02  C0 00095     ADDL2   #2, R0          ;
          50       51  D1 00098     CMPL    R1, R0          ;
          16  19 0009B     BLSS    4$              ;
          50   14   AE       01  C1 0009D     ADDL3   #1, PATTERN_STRING_SZ, R0         ; 1431
```

```
50          20   24   AE 14   AE  2C 000A2    MOVC5   PATTERN_STRING_SZ, PATTERN_BUF, #32, R0, -  ; 1432
  18   BE      000A9    @HDWRNAM_STRING_BUF_PTR       ;
  18   AE       53  D0 000AB    MOVL    R3, HDWRNAM_STRING_BUF_PTR       ;
  10   AE       01  D0 000AF    MOVL    #1, HDWRNAM_FOUND          ; 1433
       58       68  D0 000B3 4$:   MOVL    (DDB_PTR), DDB_PTR          ; 1381
       50 00000000G  EF  9E 000B6    MOVAB   MEN$GA_DDB_HEAD, R0          ;
       50       58  D1 000BD    CMPL    DDB_PTR, R0          ;
       03  13 000C0    BEQL    5$             ;
     FF78  31 000C2    BRW     2$             ;
       03 10   AE  E8 000C5 5$:   BLBS    HDWRNAM_FOUND, 6$          ; 1439
     00B5  31 000C9    BRW     15$            ;
       50 00000000'  EF  9E 000CC 6$:   MOVAB   HDWRNAM_STRING_BUF, R0         ; 1442
       50 18   AE  C2 000D3    SUBL2   HDWRNAM_STRING_BUF_PTR, R0         ;
   0C  AE       50  CE 000D7    MNEGL   R0, HDWRNAM_STRING_SZ          ;
   0C  AE       02  C2 000DB    SUBL2   #2, HDWRNAM_STRING_SZ          ;
   1C  AE  DS 000DF    TSTL    PREPTBL_INDEX          ; 1452
       06  12 000E2    BNEQ    7$             ;
   20  AE       03  C0 000E4    ADDL2   #3, SCREEN_COUNTER         ;
       3A  11 000E8    BRB     13$            ;
       01 1C   AE  D1 000EA 7$:   CMPL    PREPTBL_INDEX, #1          ; 1453
       12  13 000EE    BEQL    9$             ;
       02 1C   AE  D1 000F0    CMPL    PREPTBL_INDEX, #2          ; 1454
       06  12 000F4    BNEQ    8$             ;
   20  AE       04  C0 000F6    ADDL2   #4, SCREEN_COUNTER         ;
       28  11 000FA    BRB     13$            ;
       03 1C   AE  D1 000FC 8$:   CMPL    PREPTBL_INDEX, #3          ; 1455
       06  12 00100    BNEQ    10$            ;
   20  AE       06  C0 00102 9$:   ADDL2   #6, SCREEN_COUNTER         ;
       1C  11 00106    BRB     13$            ;
       04 1C   AE  D1 00108 10$:  CMPL    PREPTBL_INDEX, #4          ; 1456
       06  13 0010C    BEQL    11$            ;
       05 1C   AE  D1 0010E    CMPL    PREPTBL_INDEX, #5          ; 1457
       06  12 00112    BNEQ    12$            ;
   20  AE       05  C0 00114 11$:  ADDL2   #5, SCREEN_COUNTER         ;
       0A  11 00118    BRB     13$            ;
       06 1C   AE  D1 0011A 12$:  CMPL    PREPTBL_INDEX, #6          ; 1458
       04  12 0011E    BNEQ    13$            ;
   20  AE       07  C0 00120    ADDL2   #7, SCREEN_COUNTER         ;
       16 20   AE  D1 00124 13$:  CMPL    SCREEN_COUNTER, #22          ; 1461
       0D  19 00128    BLSS    14$            ;
       7E  D4 0012A    CLRL    -(SP)              ; 1464
 00000000G  EF       01  FB 0012C    CALLS   #1, CRD$SCREEN_PAUSE         ;
   20  AE       01  D0 00133    MOVL    #1, SCREEN_COUNTER          ; 1465
 00000000G  EF  9F 00137 14$:   PUSHAB  CRD$GT_NEW_LINE          ; 1476
       01  DD 0013D    PUSHL   #1             ;
       1A  DD 0013F    PUSHL   #26            ;
 00000000G  FF       03  FB 00141    CALLS   #3, @CRD$PRINT          ;
 00000000'  EF  9F 00148    PUSHAB  HDWRNAM_STRING_BUF          ; 1477
   10  AE  DD 0014E    PUSHL   HDWRNAM_STRING_SZ          ;
 00000000G  EF  9F 00151    PUSHAB  MEN$GT_DEVTSTPREP_NAME         ;
       01  DD 00157    PUSHL   #1             ;
       1A  DD 00159    PUSHL   #26            ;
 00000000G  FF       05  FB 0015B    CALLS   #5, @CRD$PRINT          ;
 00000000G  EF  9F 00162    PUSHAB  MEN$GT_DEVTSTPREP_TEXT         ; 1483
       01  DD 00168    PUSHL   #1             ;
```

```
            1A  DD 0016A     PUSHL   #26             ;
      00000000G  FF       03  FB 0016C    CALLS   #3, @CRD$PRINT          ;
      01    A7  DD 00173     PUSHL   1(ENTRY_PTR)            ; 1484
            01  DD 00176     PUSHL   #1              ;
            1A  DD 00178     PUSHL   #26             ;
      00000000G  FF       03  FB 0017A    CALLS   #3, @CRD$PRINT          ;
      1C    AE  D6 00181 15$:   INCL    PREPTBL_INDEX           ; 1360
            08 1C    AE  D1 00184    CMPL    PREPTBL_INDEX, #8          ;
            03  18 00188     BGEQ    16$             ;
          FE83  31 0018A     BRW     1$              ;
      00000000G  EF  9F 0018D 16$:   PUSHAB  MEN$GT_END_OF_LIST          ; 1489
            01  DD 00193     PUSHL   #1              ;
            1A  DD 00195     PUSHL   #26             ;
      00000000G  FF       03  FB 00197    CALLS   #3, @CRD$PRINT          ;
            03  DD 0019E     PUSHL   #3              ; 1491
      00000000G  EF       01  FB 001A0    CALLS   #1, CRD$SCREEN_PAUSE          ;
            50       01  D0 001A7    MOVL    #1, R0          ; 1494
       04 001AA     RET                 ; 1495

; Routine Size:  427 bytes,    Routine Base:  CODE + 0808
```

```
;    1496  1 %SBTTL 'MEN$Print_System_Hdwr Routine'
;    1497  1
;    1498  1 GLOBAL ROUTINE MEN$Print_System_Hdwr =
;    1499  1
;    1500  1 !++
;    1501  1 ! FUNCTIONAL DESCRIPTION:
;    1502  1 !
;    1503  1 !    Print system hardware and support status
;    1504  1 !
;    1505  1 ! INPUT PARAMETERS
;    1506  1 !
;    1507  1 !
;    1508  1 ! OUTPUT PARAMETERS
;    1509  1 !
;    1510  1 ! IMPLICIT INPUTS
;    1511  1 !
;    1512  1 ! EXPLICIT OUTPUTS
;    1513  1 !
;    1514  1 !
;    1515  1 ! SIDE EFFECTS
;    1516  1 !
;    1517  1 !
;    1518  1 ! COMPLETION CODES
;    1519  1
;    1520  1 !    CRD$K_Success
;    1521  1 ! -
```

```
;    1522  1
;    1523  2 BEGIN
;    1524  2     REGISTER
;    1525  2  Hdwr_Name_Ptr,
;    1526  2  Dev_Name,
;    1527  2  SprtStat_Ptr,
;    1528  2  DDB_Ptr  : REF Device_Data_Block_Structure,
;    1529  2  SupBlk_Ptr : REF $MEN_SupBlk_ATTR,
;    1530  2  PTable_Ptr : REF $DS_HPO_DECL ();
;    1531  2
;    1532  2    LOCAL
;    1533  2  Screen_Counter;
;    1534  2
;    1535  2     Screen_Counter = 4;   ! Init soft console line number on which
;    1536  2      ! the output will occur.
;    1537  2      ! The header title takes 1st 4 lines
;    1538  2     DDB_Ptr = MEN$GA_DDB_Head;
;    1539  2
;    1540  3     IF (.DDB_Ptr [CRD$K_DDB_Flink] NEQ MEN$GA_DDB_Head)
;    1541  2     THEN
;    1542  2  $CRD_Message (New_Line, MEN$GT_SysHdwr_Title);
;    1543  2
;    1544  2     WHILE ((DDB_Ptr = .DDB_Ptr [CRD$K_DDB_Flink]) NEQ MEN$GA_DDB Head) DO
;    1545  3  BEGIN
;    1546  3  !+
;    1547  3  ! Set up for printing Hardware Name
;    1548  3  !-
;    1549  3
;    1550  3  PTable_Ptr = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    1551  3      ! Point to the P-Table
;    1552  3  Hdwr_Name_Ptr = PTable_Ptr [HP$T_TYPE];
;    1553  3      ! Point to the Hardware device name ASCIC
;    1554  3
;    1555  3  !+
;    1556  3  ! Set up for printing Generic Device Name
;    1557  3  !-
;    1558  3
;    1559  3  Dev_Name = CRD$Get_Device_Name (.PTable_Ptr);
;    1560  3      ! Get an ASCIC device name string from the Ptable
;    1561  3
;    1562  3  !+
;    1563  3  ! Set up for printing Support Status
;    1564  3  !-
;    1565  3
;    1566  4  IF ((SupBlk_Ptr = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr]) NEQ 0)
;    1567  3  THEN
;    1568  3      SprtStat_Ptr = MEN$GT_Supported
;    1569  3  ELSE
;    1570  3      SprtStat_Ptr = MEN$GT_NoSupport;
;    1571  3      ! Point to Support Status ASCIC
;    1572  3
;    1573  3  !+
;    1574  3  ! Before printing out the line, make sure there is enough
;    1575  3  ! room on the screen.
;    1576  3  !-
```

```
;    1577  3
;    1578  4  IF (.Screen_Counter EQL CRD$K_Numb_Screen_Lines)
;    1579  4  ! AND $Soft_Console
;    1580  3  THEN
;    1581  4   BEGIN
;    1582  4   CRD$Screen_Pause (SCREEN$K_Press_Return);
;    1583  4   $CRD_Message (New_Line, MEN$GT_SysHdwr_Title);
;    1584  4   Screen_Counter = 4;
;    1585  4   END
;    1586  3  ELSE
;    1587  3   Screen_Counter = .Screen_Counter + 1;
;    1588  3
;    1589  3
;    1590  3  !+
;    1591  3  ! Print the system hardware text
;    1592  3  !-
;    1593  3
; P  1594  3  $CRD_Message ( New_Line,
; P  1595  3      MEN$GT_System_Hdwr_Text,
; P  1596  3      .Hdwr_Name_Ptr, ! Hardware Name ASCIC
; P  1597  3      .Dev_Name,  ! Generic Device Name ASCIC
; P  1598  3      .SprtStat_Ptr ! Support Status  ASCIC
;    1599  3    );
;    1600  2  END;
;    1601  2
;    1602  2     $CRD_Print (MEN$GT_End_of_List); ! Print [End of list]
;    1603  2
;    1604  2     CRD$Screen_Pause (SCREEN$K_Press_Return_MM);
;    1605  2      ! Pause before returning to MAIN MENU.
;    1606  2
;    1607  2     RETURN (CRD$K_Success);
;    1608  1 END;



        .EXTRN  MEN$GT_SYSHDWR_TITLE
        .EXTRN  MEN$GT_SUPPORTED
        .EXTRN  MEN$GT_NOSUPPORT
        .EXTRN  MEN$GT_SYSTEM_HDWR_TEXT

          01FC 00000     .ENTRY  MEN$PRINT_SYSTEM_HDWR, Save R2,R3,R4,R5,R6,-; 1498
         R7,R8                 ;
        58       04  D0 00002     MOVL    #4, SCREEN_COUNTER          ; 1535
        55 00000000G  EF  9E 00005     MOVAB   MEN$GA_DDB_HEAD, DDB_PTR          ; 1538
        50 00000000G  EF  9E 0000C     MOVAB   MEN$GA_DDB_HEAD, R0           ; 1540
        50       65  D1 00013     CMPL    (DDB_PTR), R0           ;
        03  12 00016     BNEQ    1$            ;
      00AA  31 00018     BRW     7$            ;
    00000000G  EF  9F 0001B 1$:   PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 1542
        01  DD 00021     PUSHL   #1            ;
        1A  DD 00023     PUSHL   #26            ;
    00000000G  FF       03  FB 00025     CALLS   #3, @CRD$PRINT          ;
    00000000G  EF  9F 0002C     PUSHAB  MEN$GT_SYSHDWR_TITLE          ;
        01  DD 00032     PUSHL   #1            ;
        1A  DD 00034     PUSHL   #26            ;
```

```
                00000000G  FF       03  FB 00036    CALLS   #3, @CRD$PRINT          ;
                  0085  31 0003D    BRW     7$              ; 1544
                    57 08   A5  D0 00040 2$:   MOVL    8(DDB_PTR), PTABLE_PTR          ; 1550
                    52 26   A7  9E 00044    MOVAB   38(R7), HDWR_NAME_PTR           ; 1552
                    57  DD 00048    PUSHL   PTABLE_PTR              ; 1559
                00000000G  EF       01  FB 0004A    CALLS   #1, CRD$GET_DEVICE_NAME         ;
                    53       50  D0 00051    MOVL    R0, DEV_NAME            ;
                    56 20   A5  D0 00054    MOVL    32(DDB_PTR), SUPBLK_PTR         ; 1566
                    09  13 00058    BEQL    3$              ;
                    54 00000000G  EF  9E 0005A    MOVAB   MEN$GT_SUPPORTED, SPRTSTAT_PTR          ; 1568
                    07  11 00061    BRB     4$              ;
                    54 00000000G  EF  9E 00063 3$:   MOVAB   MEN$GT_NOSUPPORT, SPRTSTAT_PTR         ; 1570
                    16       58  D1 0006A 4$:   CMPL    SCREEN_COUNTER, #22             ; 1578
                    30  12 0006D    BNEQ    5$              ;
                    7E  D4 0006F    CLRL    -(SP)               ; 1582
                00000000G  EF       01  FB 00071    CALLS   #1, CRD$SCREEN_PAUSE            ;
                00000000G  EF  9F 00078    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 1583
                    01  DD 0007E    PUSHL   #1              ;
                    1A  DD 00080    PUSHL   #26             ;
                00000000G  FF       03  FB 00082    CALLS   #3, @CRD$PRINT          ;
                00000000G  EF  9F 00089    PUSHAB  MEN$GT_SYSHDWR_TITLE            ;
                    01  DD 0008F    PUSHL   #1              ;
                    1A  DD 00091    PUSHL   #26             ;
                00000000G  FF       03  FB 00093    CALLS   #3, @CRD$PRINT          ;
                    58       04  D0 0009A    MOVL    #4, SCREEN_COUNTER          ; 1584
                    02  11 0009D    BRB     6$              ; 1578
                    58  D6 0009F 5$:   INCL    SCREEN_COUNTER          ; 1587
                00000000G  EF  9F 000A1 6$:   PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 1599
                    01  DD 000A7    PUSHL   #1              ;
                    1A  DD 000A9    PUSHL   #26             ;
                00000000G  FF       03  FB 000AB    CALLS   #3, @CRD$PRINT          ;
                    1C  BB 000B2    PUSHR   #^M<R2,R3,R4>           ;
                00000000G  EF  9F 000B4    PUSHAB  MEN$GT_SYSTEM_HDWR_TEXT         ;
                    01  DD 000BA    PUSHL   #1              ;
                    1A  DD 000BC    PUSHL   #26             ;
                00000000G  FF       06  FB 000BE    CALLS   #6, @CRD$PRINT          ;
                    55       65  D0 000C5 7$:   MOVL    (DDB_PTR), DDB_PTR          ; 1544
                    50 00000000G  EF  9E 000C8    MOVAB   MEN$GA_DDB_HEAD, R0             ;
                    50       55  D1 000CF    CMPL    DDB_PTR, R0             ;
                    03  13 000D2    BEQL    8$              ;
                  FF69  31 000D4    BRW     2$              ;
                00000000G  EF  9F 000D7 8$:   PUSHAB  MEN$GT_END_OF_LIST          ; 1602
                    01  DD 000DD    PUSHL   #1              ;
                    1A  DD 000DF    PUSHL   #26             ;
                00000000G  FF       03  FB 000E1    CALLS   #3, @CRD$PRINT          ;
                    03  DD 000E8    PUSHL   #3              ; 1604
                00000000G  EF       01  FB 000EA    CALLS   #1, CRD$SCREEN_PAUSE            ;
                    50       01  D0 000F1    MOVL    #1, R0              ; 1607
                 04 000F4    RET                 ; 1608

; Routine Size:  245 bytes,    Routine Base:  CODE + 09B3
```

```
;   1609  1 END
;   1610  0 ELUDOM



;           PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          12  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK         353  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE        2728  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



Symbol                          Type  Defined  Referenced ...
------ -------- -------------- -------- ----- ---------------
$ASCIC    Macro    Lib02   106
$CRD_LITERALS   Macro    Lib03    88
$CRD_MESSAGE   Macro    Lib03   453     462     470     553     599     720     750     760     782     790
              828     836     852    1047    1542    1583    1594
$CRD_PRINT   Unbound            453     467     470     553     599     720     750     760     782     795
              828     846     852    1047    1542    1583    1599
    Macro    Lib03   453     467     470     553     597     599     720     750     760     782
              795     828     846     852    1047    1476    1477    1483    1484    1489
             1542    1583    1599    1602
$DS_ASKSTR   KeyWMacr Lib02   512     999
$DS_BREAK   Macro    Lib02   512     999
$DS_HPO_DECL    Macro    Lib02   673    1348    1530
$DS_HPO_F   Fieldset Lib02   673    1348    1530
$DS_TYPEDEF    Macro    Lib02   453     467     470     553     597     599     720     750     760     782
              795     828     846     852    1047    1476    1477    1483    1484    1489
             1542    1583    1599    1602
$EQULST    Macro    Lib04    88      89      90     453     467     470     553     597     599     720
              750     760     782     795     828     846     852    1047    1476    1477
             1483    1484    1489    1542    1583    1599    1602
$ER    Comptime    93
$MEN_ASKSTR   KeyWMacr Lib03   509     997
$MEN_FTMTBL_ATTR  Macro    Lib03   118
$MEN_FTMTBL_ENTRY_ATTR  Macro    Lib03   430
$MEN_FTMTBL_ENTRY_FLD  Fieldset Lib03   118     430
$MEN_LITERALS   Macro    Lib03    89
$MEN_OPERINPUT_BUFFER ATTR Macro    Lib03   100     902
$MEN_PREPTBL_ATTR  Macro    Lib03   274
$MEN_PREPTBL_ENTRY_ATTR  Macro    Lib03  1346
$MEN_PREPTBL_ENTRY_FLD  Fieldset Lib03   274    1346
$MEN_QUEUE_LINK_ATTR  Macro    Lib03  1183    1249
$MEN_QUEUE_LINK_FLD  Fieldset Lib03  1183    1249
$MEN_SUPBLK_ATTR  Macro    Lib03   674    1185    1251    1349    1529
$MEN_SUPBLK_FLD   Fieldset Lib03   674    1185    1251    1349    1529
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$MEN_TMTBL_ATTR   Macro    Lib03   224
$MEN_TMTBL_ENTRY_ATTR  Macro    Lib03   656     678     888
$MEN_TMTBL_ENTRY_FLD  Fieldset Lib03   224     656     678     888
$MEN_TSTCNFG_ATTR  Macro    Lib03  1350
$MEN_TSTCNFG_FLD  Fieldset Lib03  1350
$MODULE    Bind         93
$SCREEN_LITERALS  Macro    Lib03    90
AUTO$K_EXIT_AUTOSIZER_ERROR Literal      88
AUTO$K_EXIT_CONTROL_C  Literal      88
AUTO$K_EXIT_DUMMY_2  Literal      88
AUTO$K_EXIT_DUMMY_3  Literal      88
AUTO$K_EXIT_ERRORS_COMPLETE Literal      88
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal      88
AUTO$K_EXIT_INTERNAL_ERROR Literal      88
AUTO$K_EXIT_INV_BASE_SYSTEM Literal      88
AUTO$K_EXIT_LAST_REASON  Literal      88     88
AUTO$K_EXIT_NOERRORS_COMPLETE Literal      88
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal      88
AUTO$K_EXIT_NOERRORS_PREP Literal      88
BUFFER    Bind        902   910.    913a    913.    914a
BUF_PTR    Register   891   914-    915.    915=    916.    920a.
CLASS_CODE   Bind       1103  1106.
    Bind       1188  1202.
    Bind       1254  1266.
CRD$AL_CRD_DISPATCH_VECTORS External Lib03   208a    368a
CRD$DISABLE_CTRL_C  ExtRout  Lib03   451c    718c
CRD$ENABLE_CTRL_C  ExtRout  Lib03   472c    854c
CRD$GB_CRD_DEBUG  External Lib03   604.
CRD$GET_DEVICE_NAME  ExtRout  Lib03   819c   1559c
CRD$GT_NEW_LINE   External Lib03  1476a
CRD$GT_NEW_LINE_MESSAGE  External Lib03   453a    467a    470a    553a    599a    720a    750a    760a    782a    795a
            828a    846a    852a   1047a   1542a   1583a   1599a
CRD$GT_STAR_LINE_PREFIX_MESSAGE Unbound          453     467     470     553     599     720     750     760     782     795
            828     846     852    1047    1542    1583    1599
CRD$GT_STAR_LINE_SUFFIX_MESSAGE Unbound          453     467     470     553     599     720     750     760     782     795
            828     846     852    1047    1542    1583    1599
CRD$K_ACTION_EQL_DO_NOTHING Literal      88
CRD$K_ACTION_RANGE_ERROR Literal      88
CRD$K_ADVANCE_DIAGNOSTIC Literal      88
CRD$K_ADVANCE_GENERAL_COM Literal      88
CRD$K_ADVANCE_STATE  Literal      88
CRD$K_ALLOCATION_ERROR  Literal      88
CRD$K_ASSIGN_PTABLE_PTRS Literal      88
CRD$K_AUTOSIZER_ERROR  Literal      88
CRD$K_AUTOSIZER_SET_FLAGS Literal      88
CRD$K_BAD_ACTION_NUMBER  Literal      88
CRD$K_BAD_TYPECODE_ERROR Literal      88
CRD$K_BASE_SYSTEM_IDC  Litera       88
CRD$K_BASE_SYSTEM_LES  Literal      88
CRD$K_BASE_SYSTEM_NULL  Literal      88
CRD$K_BASE_SYSTEM_UDA_0  Literal      88
CRD$K_BASE_SYSTEM_UDA_1  Literal      88
CRD$K_BASE_SYSTEM_UDA_2  Litera       88
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_BASE_SYSTEM_UDA_3  Literal     88
CRD$K_CRD_INITIALIZATION Literal     88
CRD$K_CREATE_HST  Literal     88
CRD$K_DATA_LENGTH_ERROR  Literal     88
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     88
CRD$K_DDB_BLINK   Literal     88
CRD$K_DDB_FLINK   Literal     88   806    1381    1540    1544
CRD$K_DDB_LAST_ENTRY  Literal     88
CRD$K_DDB_MEMORY_SIZE  Literal     88
CRD$K_DDB_MENU_DEVICE_NUMB Literal     88   813    1203
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     88
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     88
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     88
CRD$K_DDB_PTABLE_ENTRY  Literal     88   818    1383    1550
CRD$K_DDB_STATUS  Literal     88
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     88   807    1199    1264    1385    1566
CRD$K_DEVICE_NAME  Literal     88
CRD$K_DIAGNOSTIC_ERROR  Literal     88
CRD$K_DIAGNOSTIC_NAME  Literal     88
CRD$K_DONE_GENERAL_COMS  Literal     88
CRD$K_DO_NOTHING  Literal     88
CRD$K_DUMMY_10   Literal     88
CRD$K_DUMMY_11   Literal     88
CRD$K_DUMMY_12   Literal     88
CRD$K_DUMMY_13   Literal     88
CRD$K_DUMMY_3   Literal     88
CRD$K_DUMMY_4   Literal     88
CRD$K_DUMMY_5   Literal     88
CRD$K_DUMMY_6   Literal     88
CRD$K_DUMMY_7   Literal     88
CRD$K_DUMMY_8   Literal     88
CRD$K_DUMMY_9   Literal     88
CRD$K_EXIT_CRD   Literal     88
CRD$K_FAILURE   Literal  Lib03   615     924     951     965
CRD$K_HOOK_POINT  Literal     88
CRD$K_HS_INTERNAL_ERROR  Literal     88
CRD$K_IDC_EVDLB   Literal     88
CRD$K_IDC_EVDLC   Literal     88
CRD$K_IDC_EVDLD   Literal     88
CRD$K_IDC_EVRAD_0  Literal     88
CRD$K_IDC_EVRAD_1  Literal     88
CRD$K_IDC_LAST_DIAGNOSTIC Literal     88
CRD$K_INTERNAL_ERROR  Literal     88
CRD$K_LAST_ACTION_ROUTINE Literal     88
CRD$K_LAST_ENTRY  Literal     88
CRD$K_LAST_FUNCTION  Literal     88
CRD$K_LAST_HOOK_POINT  Literal     88
CRD$K_LAST_INTERNAL_ERROR Literal     88
CRD$K_LAST_TYPE   Literal     88
CRD$K_LESI_ENWCA  Literal     88
CRD$K_LESI_EVRMB_0  Literal     88
CRD$K_LESI_EVRMB_1  Literal     88
CRD$K_LESI_LAST_DIAGNOSTIC Literal     88
```

Symbol	Type	Defined	Referenced ...
CRD$K_LEVEL_4_PASSED	Literal	88	
CRD$K_LOAD_AUTOSIZER	Literal	88	
CRD$K_LOAD_ERROR	Literal	88	
CRD$K_LOAD_GENERAL_COM	Literal	88	
CRD$K_LOAD_LOAD_COM	Literal	88	
CRD$K_LOAD_SELECT_COM	Literal	88	
CRD$K_LOAD_SET_EVENT_COM	Literal	88	
CRD$K_LOAD_SET_FLAGS_COM	Literal	88	
CRD$K_LOAD_START_COM	Literal	88	
CRD$K_LOAD_SUP_FILE	Literal	88	
CRD$K_MM_INTERNAL_ERROR	Literal	88	
CRD$K_NEW_LINE	Literal	88	
CRD$K_NUMB_SCREEN_LINES	Literal	Lib03	746 778 824 1461 1578
CRD$K_PREPARATION_ERROR	Literal	88	
CRD$K_PREPARATION_ERROR_TEXT	Literal	88	
CRD$K_RUNTIME	Literal	88	
CRD$K_SAME_LINE	Literal	88	
CRD$K_SET_EVENT_ARGS	Literal	88	
CRD$K_SET_FLAGS_ARGS	Literal	88	
CRD$K_SET_UP_DIAGNOSTIC	Literal	88	
CRD$K_START	Literal	88	
CRD$K_START_AUTOSIZER	Literal	88	
CRD$K_START_ERROR	Literal	88	
CRD$K_START_QUALIFIERS	Literal	88	
CRD$K_STAR_LINE	Literal	88	
CRD$K_STATE_ADVANCE_DIAGNOSTIC	Literal	88	
CRD$K_STATE_ADVANCE_GENERAL_COM	Literal	88	
CRD$K_STATE_ASSIGN_PTABLE_PTRS	Literal	88	
CRD$K_STATE_AUTOSIZER_ERROR	Literal	88	
CRD$K_STATE_AUTOSIZER_RUNNING	Literal	88	
CRD$K_STATE_AUTOSIZER_SET_FLAGS	Literal	88	
CRD$K_STATE_DIAGNOSTIC_ERROR	Literal	88	
CRD$K_STATE_DIAGNOSTIC_RUNNING	Literal	88	
CRD$K_STATE_DONE_DIAGNOSTICS	Literal	88	
CRD$K_STATE_DONE_GENERAL_COMS	Literal	88	
CRD$K_STATE_DUMMY_1	Literal	88	
CRD$K_STATE_DUMMY_10	Literal	88	
CRD$K_STATE_DUMMY_12	Literal	88	
CRD$K_STATE_DUMMY_13	Literal	88	
CRD$K_STATE_DUMMY_2	Literal	88	
CRD$K_STATE_DUMMY_3	Literal	88	
CRD$K_STATE_DUMMY_4	Literal	88	
CRD$K_STATE_DUMMY_6	Literal	88	
CRD$K_STATE_DUMMY_7	Literal	88	
CRD$K_STATE_DUMMY_8	Literal	88	
CRD$K_STATE_EXIT_CRD	Literal	88	
CRD$K_STATE_INTERNAL_ERROR	Literal	88	
CRD$K_STATE_LAST_STATE	Literal	88	
CRD$K_STATE_LEVEL_4_PASSED	Literal	88	
CRD$K_STATE_LOAD_AUTOSIZER	Literal	88	
CRD$K_STATE_LOAD_ERROR	Literal	88	
CRD$K_STATE_LOAD_GENERAL_COM	Literal	88	
CRD$K_STATE_LOAD_LOAD_COM	Literal	88	

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_LOAD_SELECT_COM Literal     88
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     88
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     88
CRD$K_STATE_LOAD_START_COM Literal     88
CRD$K_STATE_PREPARATION_ERROR Literal     88
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     88
CRD$K_STATE_START  Literal     88
CRD$K_STATE_START_AUTOSIZER Literal     88
CRD$K_STATE_START_ERROR  Literal     88
CRD$K_SUCCESS   Literal  Lib03   214     374     975    1070    1145    1494    1607
CRD$K_TEST_START_TEXT  Literal     88
CRD$K_TQ_INTERNAL_ERROR  Literal     88
CRD$K_TYPECODE   Literal     88
CRD$K_UDA_0_EVDLB  Literal     88
CRD$K_UDA_0_EVDLC  Literal     88
CRD$K_UDA_0_EVDLD  Literal     88
CRD$K_UDA_0_EVRLA_0  Literal     88
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     88
CRD$K_UDA_1_EVDLB  Literal     88
CRD$K_UDA_1_EVDLC  Literal     88
CRD$K_UDA_1_EVDLD  Literal     88
CRD$K_UDA_1_EVRLA_0  Literal     88
CRD$K_UDA_1_EVRLA_1  Literal     88
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     88
CRD$K_UDA_2_EVDLB  Literal     88
CRD$K_UDA_2_EVDLC  Literal     88
CRD$K_UDA_2_EVDLD  Literal     88
CRD$K_UDA_2_EVRLA_0  Literal     88
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     88
CRD$K_UDA_3_EVDLB  Literal     88
CRD$K_UDA_3_EVDLC  Literal     88
CRD$K_UDA_3_EVDLD  Literal     88
CRD$K_UDA_3_EVRLA_0  Literal     88
CRD$K_UDA_3_EVRLA_1  Literal     88
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     88
CRD$PRINT   External Lib03   453ac   467ac   470ac   553ac   597ac   599ac   720ac   750ac   760ac   782ac
             795ac   828ac   846ac   852ac  1047ac  1476ac  1477ac  1483ac  1488ac  1489ac
            1542ac  1583ac  1599ac  1602ac
CRD$SCREEN_PAUSE  ExtRout  Lib03   519c    749c    781c    827c   1005c   1464c   1491c   1582c   1604c
CRD$STOP   ExtRout  Lib03   564c
DDB    Bind      1190  1208=   1210=
    Bind      1255  1271=   1273=
DDB_FOUND   Register  1182  1192=   1195.   1201=   1206.   1212.
    Register  1248  1257=   1260.   1266=   1269.   1274.
DDB_PTR    Local      657  1032a   1067.
    Local      677   703a    805=    806=    806a.   807a.   813a.   818a.
    Register  1184  1197=   1199a.  1203a.  1208.
    Register  1250  1262=   1264a.  1271.
    Register  1347  1379=   1381=   1381a.  1383a.  1385a.
    Register  1528  1538=   1540a.  1544=   1544a.  1550a.  1566a.
DEVICE_DATA_BLOCK_STRUCTURE Structur Lib03   657     677    1184    1250    1347    1528
DEVICE_NUMBER   Local      655  1011a   1021.   1031a
DEVTSTPREP_CODE   Local     1334  1363=   1410.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DEV_NAME   Local       686   819=    846.
    Register  1526  1559=   1599.
DS$ASKSTR   ExtRout     512   512c    999e    999c
DS$BREAK   ExtRout     512   512c    512e    999e    999c
DS$K_PRINTB   Literal     453
    Literal     453
    Literal     467
    Literal     467
    Literal     470
    Literal     470
    Literal     553
    Literal     553
    Literal     597
    Literal     599
    Literal     599
    Literal     720
    Literal     720
    Literal     750
    Literal     750
    Literal     760
    Literal     760
    Literal     782
    Literal     782
    Literal     795
    Literal     795
    Literal     828
    Literal     828
    Literal     828
    Literal     846
    Literal     846
    Literal     852
    Literal     852
    Literal    1047
    Literal    1047
    Literal    1476
    Literal    1477
    Literal    1483
    Literal    1484
    Literal    1489
    Literal    1542
    Literal    1542
    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
DS$K_PRINTF   Literal     453   453
    Literal     453   453
    Literal     467   467
    Literal     467   467
    Literal     470   470
    Literal     470   470
    Literal     553   553
    Literal     553   553
```

Symbol	Type	Defined	Referenced ...
	Literal	597	597
	Literal	599	599
	Literal	599	599
	Literal	720	720
	Literal	720	720
	Literal	750	750
	Literal	750	750
	Literal	760	760
	Literal	760	760
	Literal	782	782
	Literal	782	782
	Literal	795	795
	Literal	795	795
	Literal	828	828
	Literal	828	828
	Literal	846	846
	Literal	846	846
	Literal	852	852
	Literal	852	852
	Literal	1047	1047
	Literal	1047	1047
	Literal	1476	1476
	Literal	1477	1477
	Literal	1483	1483
	Literal	1484	1484
	Literal	1489	1489
	Literal	1542	1542
	Literal	1542	1542
	Literal	1583	1583
	Literal	1583	1583
	Literal	1599	1599
	Literal	1599	1599
	Literal	1602	1602
DS$K_PRINTI	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	

Symbol	Type	Defined	Referenced ...
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_PRINTX	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    1484
    Literal    1489
    Literal    1542
    Literal    1542
    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
DS$K_TYPE_ABORT_PROGRAM  Literal     453
    Literal     453
    Literal     467
    Literal     467
    Literal     470
    Literal     470
    Literal     553
    Literal     553
    Literal     597
    Literal     599
    Literal     599
    Literal     720
    Literal     720
    Literal     750
    Literal     750
    Literal     760
    Literal     760
    Literal     782
    Literal     782
    Literal     795
    Literal     795
    Literal     828
    Literal     828
    Literal     846
    Literal     846
    Literal     852
    Literal     852
    Literal    1047
    Literal    1047
    Literal    1476
    Literal    1477
    Literal    1483
    Literal    1484
    Literal    1489
    Literal    1542
    Literal    1542
    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
DS$K_TYPE_ABORT_TEST  Literal     453
    Literal     453
    Literal     467
```

Symbol	Type	Defined	Referenced ...
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_COMMAND_ERR	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	

Symbol	Type	Defined	Referenced ...
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_COMMAND_OUT	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	

Symbol	Type	Defined	Referenced ...
Literal	1047		
Literal	1047		
Literal	1476		
Literal	1477		
Literal	1483		
Literal	1484		
Literal	1489		
Literal	1542		
Literal	1542		
Literal	1583		
Literal	1583		
Literal	1599		
Literal	1599		
Literal	1602		
DS$K_TYPE_CRD_AUTOTEST	Literal	453	453
Literal	453	453	
Literal	467	467	
Literal	467	467	
Literal	470	470	
Literal	470	470	
Literal	553	553	
Literal	553	553	
Literal	597	597	
Literal	599	599	
Literal	599	599	
Literal	720	720	
Literal	720	720	
Literal	750	750	
Literal	750	750	
Literal	760	760	
Literal	760	760	
Literal	782	782	
Literal	782	782	
Literal	795	795	
Literal	795	795	
Literal	828	828	
Literal	828	828	
Literal	846	846	
Literal	846	846	
Literal	852	852	
Literal	852	852	
Literal	1047	1047	
Literal	1047	1047	
Literal	1476	1476	
Literal	1477	1477	
Literal	1483	1483	
Literal	1484	1484	
Literal	1489	1489	
Literal	1542	1542	
Literal	1542	1542	
Literal	1583	1583	
Literal	1583	1583	
Literal	1599	1599	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal   1599  1599
    Literal   1602  1602
DS$K_TYPE_DS_PROMPT  Literal    453
    Literal    453
    Literal    467
    Literal    467
    Literal    470
    Literal    470
    Literal    553
    Literal    553
    Literal    597
    Literal    599
    Literal    599
    Literal    720
    Literal    720
    Literal    750
    Literal    750
    Literal    760
    Literal    760
    Literal    782
    Literal    782
    Literal    795
    Literal    795
    Literal    828
    Literal    828
    Literal    846
    Literal    846
    Literal    852
    Literal    852
    Literal   1047
    Literal   1047
    Literal   1476
    Literal   1477
    Literal   1483
    Literal   1484
    Literal   1489
    Literal   1542
    Literal   1542
    Literal   1583
    Literal   1583
    Literal   1599
    Literal   1599
    Literal   1602
DS$K_TYPE_DS_START  Literal    453
    Literal    453
    Literal    467
    Literal    467
    Literal    470
    Literal    470
    Literal    553
    Literal    553
    Literal    597
    Literal    599
```

Symbol	Type	Defined	Referenced ...
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_ERRDEV	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	

Symbol	Type	Defined	Referenced ...
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_ERRHARD	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	

Symbol	Type	Defined	Referenced ...
	Literal		1542
	Literal		1542
	Literal		1583
	Literal		1583
	Literal		1599
	Literal		1599
	Literal		1602
DS$K_TYPE_ERROR_BODY	Literal		453
	Literal		453
	Literal		467
	Literal		467
	Literal		470
	Literal		470
	Literal		553
	Literal		553
	Literal		597
	Literal		599
	Literal		599
	Literal		720
	Literal		720
	Literal		750
	Literal		750
	Literal		760
	Literal		760
	Literal		782
	Literal		782
	Literal		782
	Literal		795
	Literal		795
	Literal		828
	Literal		828
	Literal		846
	Literal		846
	Literal		852
	Literal		852
	Literal		1047
	Literal		1047
	Literal		1476
	Literal		1477
	Literal		1483
	Literal		1484
	Literal		1489
	Literal		1542
	Literal		1542
	Literal		1583
	Literal		1583
	Literal		1599
	Literal		1599
	Literal		1602
DS$K_TYPE_ERROR_END	Literal		453
	Literal		453
	Literal		467
	Literal		467
	Literal		470

Symbol	Type	Defined	Referenced ...
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_ERRPREP	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     782
    Literal     782
    Literal     795
    Literal     795
    Literal     828
    Literal     828
    Literal     846
    Literal     846
    Literal     852
    Literal     852
    Literal    1047
    Literal    1047
    Literal    1476
    Literal    1477
    Literal    1483
    Literal    1484
    Literal    1489
    Literal    1542
    Literal    1542
    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
DS$K_TYPE_ERRSOFT  Literal     453
    Literal     453
    Literal     467
    Literal     467
    Literal     467
    Literal     470
    Literal     470
    Literal     553
    Literal     553
    Literal     597
    Literal     599
    Literal     599
    Literal     720
    Literal     720
    Literal     750
    Literal     750
    Literal     760
    Literal     760
    Literal     782
    Literal     782
    Literal     795
    Literal     795
    Literal     828
    Literal     828
    Literal     846
    Literal     846
    Literal     852
    Literal     852
    Literal    1047
    Literal    1047
```

Symbol	Type	Defined	Referenced ...
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_ERRSUP	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	

Symbol	Type	Defined	Referenced ...
DS$K_TYPE_ERRSYS	Literal	453	
	Literal		453
	Literal		467
	Literal		467
	Literal		470
	Literal		470
	Literal		553
	Literal		553
	Literal		597
	Literal		599
	Literal		599
	Literal		720
	Literal		720
	Literal		750
	Literal		750
	Literal		760
	Literal		760
	Literal		782
	Literal		782
	Literal		795
	Literal		795
	Literal		828
	Literal		828
	Literal		846
	Literal		846
	Literal		852
	Literal		852
	Literal		1047
	Literal		1047
	Literal		1476
	Literal		1477
	Literal		1483
	Literal		1484
	Literal		1489
	Literal		1542
	Literal		1542
	Literal		1583
	Literal		1583
	Literal		1599
	Literal		1599
	Literal		1602
DS$K_TYPE_ERR_HALT	Literal	453	
	Literal		453
	Literal		467
	Literal		467
	Literal		470
	Literal		470
	Literal		553
	Literal		553
	Literal		597
	Literal		599
	Literal		599
	Literal		720

```
Symbol                          Type  Defined  Referenced ...
- ------------------------------ ------- ----- ---------------
    Literal     720
    Literal     750
    Literal     750
    Literal     760
    Literal     760
    Literal     782
    Literal     782
    Literal     795
    Literal     795
    Literal     828
    Literal     828
    Literal     846
    Literal     846
    Literal     852
    Literal     852
    Literal    1047
    Literal    1047
    Literal    1476
    Literal    1477
    Literal    1483
    Literal    1484
    Literal    1489
    Literal    1542
    Literal    1542
    Literal    1583
    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
DS$K_TYPE_EXCEPTION  Literal     453
    Literal     453
    Literal     467
    Literal     467
    Literal     470
    Literal     470
    Literal     553
    Literal     553
    Literal     597
    Literal     599
    Literal     599
    Literal     720
    Literal     720
    Literal     750
    Literal     750
    Literal     760
    Literal     760
    Literal     782
    Literal     782
    Literal     795
    Literal     795
    Literal     828
    Literal     828
    Literal     846
```

Symbol	Type	Defined	Referenced ...
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_EXCEPTION_HEAD	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------

    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
DS$K_TYPE_FIRST_PASS  Literal    453
    Literal     453
    Literal     467
    Literal     467
    Literal     470
    Literal     470
    Literal     553
    Literal     553
    Literal     597
    Literal     599
    Literal     599
    Literal     720
    Literal     720
    Literal     750
    Literal     750
    Literal     760
    Literal     760
    Literal     782
    Literal     782
    Literal     795
    Literal     795
    Literal     828
    Literal     828
    Literal     828
    Literal     846
    Literal     846
    Literal     852
    Literal     852
    Literal    1047
    Literal    1047
    Literal    1476
    Literal    1477
    Literal    1483
    Literal    1484
    Literal    1489
    Literal    1542
    Literal    1542
    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
DS$K_TYPE_GENERAL  Literal    453
    Literal     453
    Literal     467
    Literal     467
    Literal     470
    Literal     470
    Literal     553
```

Symbol	Type	Defined	Referenced ...
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_GENERAL_ERROR	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	

Symbol	Type	Defined	Referenced ...
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_NO_TESTS	Literal	453	
	Litera	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	

Symbol	Type	Defined	Referenced ...
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_PARAM_ERROR	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_PROGRAM_END	Literal	453	
	Literal	453	

Symbol	Type	Defined	Referenced ...
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_PROGRAM_INFO	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	

Symbol	Type	Defined	Referenced ...
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_PROGRAM_START	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	

Symbol	Type	Defined	Referenced ...
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_QIO_INVADP	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------

    Literal   1599
    Literal   1599
    Literal   1602
DS$K_TYPE_QIO_NODRIVER  Literal    453
    Literal    453
    Literal    467
    Literal    467
    Literal    470
    Literal    470
    Literal    553
    Literal    553
    Literal    597
    Literal    599
    Literal    599
    Literal    720
    Literal    720
    Literal    750
    Literal    750
    Literal    760
    Literal    760
    Literal    782
    Literal    782
    Literal    795
    Literal    795
    Literal    828
    Literal    828
    Literal    846
    Literal    846
    Literal    852
    Literal    852
    Literal   1047
    Literal   1047
    Literal   1476
    Literal   1477
    Literal   1483
    Literal   1484
    Literal   1489
    Literal   1542
    Literal   1542
    Literal   1583
    Literal   1583
    Literal   1599
    Literal   1599
    Literal   1602
DS$K_TYPE_QIO_WRONGVER  Literal    453
    Literal    453
    Literal    467
    Literal    467
    Literal    470
    Literal    470
    Literal    553
    Literal    553
    Literal    597
```

Symbol	Type	Defined	Referenced ...
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_SCRIPT_ECHO	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	

Symbol	Type	Defined	Referenced ...
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_SCRIPT_PNF	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    1489
    Literal    1542
    Literal    1542
    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
DS$K_TYPE_SCRIPT_PROMPT  Literal     453
    Literal     453
    Literal     467
    Literal     467
    Literal     470
    Literal     470
    Literal     553
    Literal     553
    Literal     597
    Literal     599
    Literal     599
    Literal     720
    Literal     720
    Literal     750
    Literal     750
    Literal     760
    Literal     760
    Literal     760
    Literal     782
    Literal     782
    Literal     795
    Literal     795
    Literal     828
    Literal     828
    Literal     846
    Literal     846
    Literal     852
    Literal     852
    Literal    1047
    Literal    1047
    Literal    1476
    Literal    1477
    Literal    1483
    Literal    1484
    Literal    1489
    Literal    1542
    Literal    1542
    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
DS$K_TYPE_SCRIPT_SKIP  Literal     453
    Literal     453
    Literal     467
    Literal     467
```

Symbol	Type	Defined	Referenced ...
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	
	Literal	760	
	Literal	782	
	Literal	782	
	Literal	795	
	Literal	795	
	Literal	828	
	Literal	828	
	Literal	846	
	Literal	846	
	Literal	852	
	Literal	852	
	Literal	1047	
	Literal	1047	
	Literal	1476	
	Literal	1477	
	Literal	1483	
	Literal	1484	
	Literal	1489	
	Literal	1542	
	Literal	1542	
	Literal	1583	
	Literal	1583	
	Literal	1599	
	Literal	1599	
	Literal	1602	
DS$K_TYPE_SEQUENCE_ERROR	Literal	453	
	Literal	453	
	Literal	467	
	Literal	467	
	Literal	470	
	Literal	470	
	Literal	553	
	Literal	553	
	Literal	597	
	Literal	599	
	Literal	599	
	Literal	720	
	Literal	720	
	Literal	750	
	Literal	750	
	Literal	760	

Symbol	Type	Defined	Referenced ...
	Literal		760
	Literal		782
	Literal		782
	Literal		795
	Literal		795
	Literal		828
	Literal		828
	Literal		846
	Literal		846
	Literal		852
	Literal		852
	Literal		1047
	Literal		1047
	Literal		1476
	Literal		1477
	Literal		1483
	Literal		1484
	Literal		1489
	Literal		1542
	Literal		1542
	Literal		1583
	Literal		1583
	Literal		1599
	Literal		1599
	Literal		1602
DS$K_TYPE_START_ERR	Literal		453
	Literal		453
	Literal		467
	Literal		467
	Literal		470
	Literal		470
	Literal		553
	Literal		553
	Literal		597
	Literal		599
	Literal		599
	Literal		720
	Literal		720
	Literal		750
	Literal		750
	Literal		760
	Literal		760
	Literal		782
	Literal		782
	Literal		795
	Literal		795
	Literal		828
	Literal		828
	Literal		846
	Literal		846
	Literal		852
	Literal		852
	Literal		1047

Symbol	Type	Defined	Referenced ...
Literal 1047			
Literal 1476			
Literal 1477			
Literal 1483			
Literal 1484			
Literal 1489			
Literal 1542			
Literal 1542			
Literal 1583			
Literal 1583			
Literal 1599			
Literal 1599			
Literal 1602			
DS$K_TYPE_START_LIST	Literal	453	
Literal 453			
Literal 467			
Literal 467			
Literal 470			
Literal 470			
Literal 553			
Literal 553			
Literal 597			
Literal 599			
Literal 599			
Literal 720			
Literal 720			
Literal 750			
Literal 750			
Literal 750			
Literal 760			
Literal 760			
Literal 782			
Literal 782			
Literal 795			
Literal 795			
Literal 828			
Literal 828			
Literal 846			
Literal 846			
Literal 852			
Literal 852			
Literal 1047			
Literal 1047			
Literal 1476			
Literal 1477			
Literal 1483			
Literal 1484			
Literal 1489			
Literal 1542			
Literal 1542			
Literal 1583			
Literal 1583			
Literal 1599			
Literal 1599			

```
Symbol                          Type  Defined  Referenced ...
------ ------------------------ -------- ----- ---------------

    Literal   1602
DS$K_TYPE_SUMMARY  Literal    453
    Literal    453
    Literal    467
    Literal    467
    Literal    470
    Literal    470
    Literal    553
    Literal    553
    Literal    597
    Literal    599
    Literal    599
    Literal    720
    Literal    720
    Literal    750
    Literal    750
    Literal    760
    Literal    760
    Literal    782
    Literal    782
    Literal    795
    Literal    795
    Literal    828
    Literal    828
    Literal    846
    Literal    846
    Literal    852
    Literal    852
    Literal    852
    Literal   1047
    Literal   1047
    Literal   1476
    Literal   1477
    Literal   1483
    Literal   1484
    Literal   1489
    Literal   1542
    Literal   1542
    Literal   1583
    Literal   1583
    Literal   1599
    Literal   1599
    Literal   1602
DS$K_TYPE_USER_PROMPT  Literal    453
    Literal    453
    Literal    467
    Literal    467
    Literal    470
    Literal    470
    Literal    553
    Literal    553
    Literal    597
    Literal    599
    Literal    599
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     720
    Literal     720
    Literal     750
    Literal     750
    Literal     760
    Literal     760
    Literal     782
    Literal     782
    Literal     795
    Literal     795
    Literal     828
    Literal     828
    Literal     846
    Literal     846
    Literal     852
    Literal     852
    Literal    1047
    Literal    1047
    Literal    1476
    Literal    1477
    Literal    1483
    Literal    1484
    Literal    1489
    Literal    1542
    Literal    1542
    Literal    1583
    Literal    1583
    Literal    1583
    Literal    1599
    Literal    1599
    Literal    1602
ENTRY    Bind       903   944=
ENTRY_PTR   Local      430   532=    533a.   561a.
    Local       656  1011a   1013a.  1030a.  1055a.
    Local       678   701=    703a.   704a.   707a=   711a=   735=    737a.   740a.   768a.   771.
               808a.   812.    816a.
    Local       888   936-    944.
    Register   1346  1362=   1363a.  1484a.
FALSE    Literal  Lib03   180      340      495      613      932      991     1192     1257     1377
FTMTBL$B_ENTRY_ALPHA  Field    Lib03   459a    532a    533.
FTMTBL$B_ENTRY_FTMSEL  Field    Lib03   561.
FTMTBL$L_ENTRY_TEXT  Field    Lib03   206a    460.
FTMTBL_INIT   Own         180   180=    195.    211=
FTM_TABLE_INDEX   Local       431   526=    528=    528.    532.
    Register    447   455=    457=    457.    459.    460.
GET1ST_    Macro    Lib04    88       89       90      453      467      470      553      597      599      720
              750      760      782      795      828      846      852     1047     1476     1477
             1483     1484     1489     1542     1583     1599     1602
GET2ND_    Unbound             88       89       90      453      467      470      553      597      599      720
              750      760      782      795      828      846      852     1047     1476     1477
             1483     1484     1489     1542     1583     1599     1602
HDWRNAM_FOUND   Local      1335  1377=   1433=   1439.
HDWRNAM_PTR   Register  1345  1396=   1399a.  1419a.
HDWRNAM_STRING_BUF  Bind        103  1364a   1366a   1414.   1442a   1477a
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------- -------- ----- ---------------
HDWRNAM_STRING_BUF_END  Local      1337  1366=   1419.
HDWRNAM_STRING_BUF_PTR  Local      1336  1364=   1419.   1427=   1432a=  1442.
HDWRNAM_STRING_BUF_SZ  Literal      97  1366    1413
HDWRNAM_STRING_SZ  Local      1338  1442=   1477.
HDWR_NAME_PTR   Local       685   814=    846.
    Register  1525  1552=   1599.
HP$K_LENGTH   Literal  Lib02   673    1348    1530
HP$T_TYPE   Field    Lib02  1396a   1552a
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal      88
HS$K_ACTION_ADVANCE_GENERAL_COM Literal      88
HS$K_ACTION_ADVANCE_STATE Literal      88
HS$K_ACTION_CHANGE_TABLE Literal      88
HS$K_ACTION_DIAGNOSTIC_ERROR Literal      88
HS$K_ACTION_DONE_GENERAL_COMS Literal      88
HS$K_ACTION_DO_NOTHING  Literal      88
HS$K_ACTION_DUMMY_3  Literal      88
HS$K_ACTION_DUMMY_4  Literal      88
HS$K_ACTION_DUMMY_5  Literal      88
HS$K_ACTION_DUMMY_6  Literal      88
HS$K_ACTION_INIT_HS  Literal      88
HS$K_ACTION_INTERNAL_ERROR Literal      88
HS$K_ACTION_LAST_ACTION  Literal      88
HS$K_ACTION_LOAD_ERROR  Literal      88
HS$K_ACTION_LOAD_GENERAL_COM Literal      88
HS$K_ACTION_LOAD_LOAD_COM Literal      88
HS$K_ACTION_LOAD_SELECT_COM Literal      88
HS$K_ACTION_LOAD_SET_EVENT_COM Literal      88
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal      88
HS$K_ACTION_LOAD_START_COM Literal      88
HS$K_ACTION_PREPARATION_ERROR Literal      88
HS$K_ACTION_START_ERROR  Literal      88
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal      88
HS$K_STATE_ADVANCE_GENERAL_COM Literal      88
HS$K_STATE_CHANGE_TABLE  Literal      88
HS$K_STATE_DIAGNOSTIC_ERROR Literal      88
HS$K_STATE_DIAGNOSTIC_RUNNING Litera      88
HS$K_STATE_DONE_GENERAL_COMS Literal      88
HS$K_STATE_DUMMY_1  Literal      88
HS$K_STATE_DUMMY_2  Literal      88
HS$K_STATE_DUMMY_3  Literal      88
HS$K_STATE_DUMMY_4  Literal      88
HS$K_STATE_DUMMY_6  Literal      88
HS$K_STATE_INIT_HS  Literal      88
HS$K_STATE_INTERNAL_ERROR Literal      88
HS$K_STATE_LAST_STATE  Literal      88
HS$K_STATE_LOAD_ERROR  Literal      88
HS$K_STATE_LOAD_GENERAL_COM Literal      88
HS$K_STATE_LOAD_LOAD_COM Literal      88
HS$K_STATE_LOAD_SELECT_COM Literal      88
HS$K_STATE_LOAD_SET_EVENT_COM Literal      88
HS$K_STATE_LOAD_SET_FLAGS_COM Literal      88
HS$K_STATE_LOAD_START_COM Literal      88
HS$K_STATE_PREPARATION_ERROR Literal      88
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_STATE_START_ERROR  Literal     88
INDEX    Local      177   202=    204=    204.    206.
    Local      337   362=    364=    364.    366.
INPUT_ALPHA   Register   892   915=    940.    941.
INPUT_END   Local      886   913=    916.
I_BUFFER   RoutForm   880   902.
I_DDB    RoutForm  1149  1190.
    RoutForm  1216  1255.
I_ENTRY    RoutForm   880   903.
I_NAME    RoutForm  1074  1104.
I_NUMBER   RoutForm   880   904.
LAST_TSTCLA   Local      681   728=    740.
MEN$BLDTSTQ_FNCTST_ALL  ExtRout  Lib03   586c   1061c
MEN$BLDTSTQ_FNCTST_DEV  ExtRout  Lib03  1067c
MEN$DUMP_TSTQ_HST  ExtRout  Lib03   606c
MEN$FNDDDB_TCL   GlobRout  1216 Lib03e     73f    703c
MEN$FNDDDB_TCL_SDN  GlobRout  1149 Lib03e     72f   1029c
MEN$FTMTBL_INIT   GlobRout   143 Lib03e     68f
MEN$FUNCTEST_MENU  GlobRout   378 Lib03e     67f
MEN$GA_DDB_HEAD   External Lib03   805a    806    1193a   1195    1258a   1260    1379a   1381    1538a   1540
          1544
MEN$GA_FTMTBL   Global     118 Lib03e    119=    206a    459a    460.    532a
MEN$GA_PREPTBL   Global     274 Lib03e    275-    366a   1362a
MEN$GA_TMTBL   Global     224 Lib03e    225=    701a    735a    936a    937.
MEN$GB_FNCTST_INIT  External Lib03   480=
MEN$GB_TEST_IN_PROGRESS  External Lib03   602=
MEN$GET_TST_A_NAME  GlobRout  1074 Lib03e     71f    815c
MEN$GK_FTMTBL_ENTRIES  Literal  Lib03   118     204     457     528
MEN$GK_PREPTBL_ENTRIES  Literal  Lib03   274     364    1360
MEN$GK_TEST_QUEUE  Literal     89
MEN$GK_TMTBL_ENTRIES  Literal  Lib03   224     699     733     934
MEN$GT_CTRLC_FNCTST  External Lib03   597a
MEN$GT_DEVTSTPREP_1  External Lib03   281a
MEN$GT_DEVTSTPREP_2  External Lib03   284a
MEN$GT_DEVTSTPREP_3  External Lib03   287a
MEN$GT_DEVTSTPREP_4  External Lib03   290a
MEN$GT_DEVTSTPREP_5  External Lib03   293a
MEN$GT_DEVTSTPREP_6  External Lib03   296a
MEN$GT_DEVTSTPREP_7  External Lib03   299a
MEN$GT_DEVTSTPREP_NAME  External Lib03  1477a
MEN$GT_DEVTSTPREP_NULL  External Lib03   278a
MEN$GT_DEVTSTPREP_TEXT  External Lib03  1483a
MEN$GT_END_OF_LIST  External Lib03  1489a   1602a
MEN$GT_FTMENU_SELECTION  External Lib03   467a
MEN$GT_FTMPROMPT  External Lib03   512a
MEN$GT_FTMSEL_EXIT  External Lib03   123a
MEN$GT_FTMSEL_FNCTST_ALL External Lib03   139a
MEN$GT_FTMSEL_PREP  External Lib03   131a
MEN$GT_FTMSEL_SYSTEM_HDWR External Lib03   127a
MEN$GT_FTMSEL_TEST  External Lib03   135a
MEN$GT_FTMTITLE   External Lib03   453a
MEN$GT_FTM_CHOICE  External Lib03   470a
MEN$GT_FUNCTEST_BEGIN  External Lib03   599a
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$GT_INVOPERINPUT  External Lib03   553a   1047a
MEN$GT_NOSUPPORT  External Lib03  1570a
MEN$GT_NULL   External Lib03   760a
MEN$GT_OPERINPUT_BUFFER  Global     100 Lib03e    103a    512a    537.    538.    550.    999a   1011a   1044.
MEN$GT_SUPPORTED  External Lib03  1568a
MEN$GT_SYSHDWR_TITLE  External Lib03  1542a   1583a
MEN$GT_SYSTEM_HDWR_TEXT  External Lib03  1599a
MEN$GT_TMENU_SELECTION  External Lib03   846a
MEN$GT_TMENU_TSTALL  External Lib03   795a
MEN$GT_TMPROMPT   External Lib03   999a
MEN$GT_TMTITLE   External Lib03   720a    750a    782a    828a
MEN$GT_TM_CHOICE  External Lib03   852a
MEN$GT_TSTCLA_ALL  External Lib03  1139a
MEN$GT_TSTCLA_CARD  External Lib03  1133a
MEN$GT_TSTCLA_COMM  External Lib03  1121a
MEN$GT_TSTCLA_CPU  External Lib03  1109a
MEN$GT_TSTCLA_DISK  External Lib03  1115a
MEN$GT_TSTCLA_IOADAP  External Lib03  1136a
MEN$GT_TSTCLA_MEMORY  External Lib03  1112a
MEN$GT_TSTCLA_NULLNAME  External Lib03  1142a
MEN$GT_TSTCLA_PRINTER  External Lib03  1130a
MEN$GT_TSTCLA_REAL  External Lib03  1124a
MEN$GT_TSTCLA_TAPE  External Lib03  1118a
MEN$GT_TSTCLA_TERM  External Lib03  1127a
MEN$K_CNTLC_SEL_CONTINUE Literal     89
MEN$K_CNTLC_SEL_EXIT  Literal     89
MEN$K_CNTLC_SEL_FTMENU  Literal     89
MEN$K_CNTLC_SEL_LAST_ENTRY Literal     89
MEN$K_DEVTSTPREP_1  Literal     89   280
MEN$K_DEVTSTPREP_2  Literal     89   283
MEN$K_DEVTSTPREP_3  Literal     89   286
MEN$K_DEVTSTPREP_4  Literal     89   289
MEN$K_DEVTSTPREP_5  Literal     89   292
MEN$K_DEVTSTPREP_6  Literal     89   295
MEN$K_DEVTSTPREP_7  Literal     89   298
MEN$K_DEVTSTPREP_NULL  Litera       89   277
MEN$K_FTMRET_EXIT  Literal     89
MEN$K_FTMRET_FAILURE  Literal     89   579
MEN$K_FTMRET_MENU  Literal     89
MEN$K_FTMRET_TEST  Literal     89   608
MEN$K_FTMSEL_EXIT  Literal     89   122     563
MEN$K_FTMSEL_FNCTST_ALL  Literal     89   138     582     588
MEN$K_FTMSEL_LAST_ENTRY  Literal     89
MEN$K_FTMSEL_PREP  Literal     89   130     569
MEN$K_FTMSEL_SYSTEM_HDWR Literal     89   126     566
MEN$K_FTMSEL_TEST  Literal     89   134     572     588
MEN$K_FTMTBL_ENTRY_SIZE  Literal  Lib03   118     430
MEN$K_HS_STATE_TABLE  Literal     88
MEN$K_MM_STATE_TABLE  Literal     88
MEN$K_OPERINPUT_BUFFER_SIZE Literal  Lib03    97     100     902
MEN$K_PREPTBL_ENTRY_SIZE Literal  Lib03   274    1346
MEN$K_PREP_ERROR_TEXT_LEN Literal     89
MEN$K_QUEUE_LINK_SIZE  Literal  Lib03  1183    1249
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$K_SELDEV_NUMB_START  Literal     89
MEN$K_SUPBLK_SIZE  Literal  Lib03   674    1185    1251    1349    1529
MEN$K_TMENU_TSTALL_DEVNUMB Literal     89   772    1021
MEN$K_TMTBL_ENTRY_SIZE  Literal  Lib03   224     656     678     888
MEN$K_TQ_STATE_TABLE  Literal     88
MEN$K_TSTCLA_ALL  Literal     89   260     704     768    1015    1057    1138
MEN$K_TSTCLA_CARD  Literal     89   254    1132
MEN$K_TSTCLA_COMM  Literal     89   242    1120
MEN$K_TSTCLA_CPU  Literal     89   230    1108
MEN$K_TSTCLA_DISK  Literal     89   236    1114
MEN$K_TSTCLA_IO_ADAPTOR  Literal     89   257    1135
MEN$K_TSTCLA_LAST_ENTRY  Literal     89
MEN$K_TSTCLA_MEMORY  Literal     89   233    1111
MEN$K_TSTCLA_NULL  Literal     89   728
MEN$K_TSTCLA_PRINTER  Literal     89   251    1129
MEN$K_TSTCLA_REAL  Literal     89   245    1123
MEN$K_TSTCLA_TAPE  Literal     89   239    1117
MEN$K_TSTCLA_TERM  Literal     89   248    1126
MEN$K_TSTCNFG_SIZE  Literal  Lib03  1350
MEN$K_TSTCTG_CPU  Literal     89
MEN$K_TSTCTG_IO_ADAPTOR  Literal     89
MEN$K_TSTCTG_IO_CONTROLLER Literal     89
MEN$K_TSTCTG_IO_UNIT  Literal     89
MEN$K_TSTCTG_MEMORY  Literal     89
MEN$K_TSTCTG_NULL  Literal     89
MEN$K_TSTTXT_DEVNAM_SZ  Literal     89
MEN$K_TSTTXT_TSTCLA_SZ  Literal     89
MEN$K_TSTTXT_UUTNAM_SZ  Literal     89
MEN$PREPTBL_INIT  GlobRout   303 Lib03e     70f
MEN$PRINT_DEVTSTPREP  GlobRout  1278 Lib03e     74f     570c
MEN$PRINT_RUNTIME_TOTAL  ExtRout  Lib03   601c
MEN$PRINT_SYSTEM_HDWR  GlobRout  1498 Lib03e     75f     567c
MEN$SCAN_NUMERIC_ASC_D  ExtRout  Lib03   973c
MEN$SCRATCH_MEDIA_MSG  ExtRout  Lib03   593c
MEN$TEST_MENU  GlobRout   619 Lib03e     69f     577c
MENU$K_EXIT_AUTOSIZER_ERROR Literal     88
MENU$K_EXIT_INTERNAL_ERROR Literal     88
MENU$K_EXIT_LAST_REASON  Literal     88
MENU$K_EXIT_OPERATOR_ABORT Literal     88
MENU$K_EXIT_OPERATOR_STOP Literal     88   564
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     88
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     88
MM$K_ACTION_ADVANCE_STATE Literal     88
MM$K_ACTION_AUTOSIZER_ERROR Literal     88
MM$K_ACTION_BUILD_DDBS  Literal     88
MM$K_ACTION_BUILD_HSTS  Literal     88
MM$K_ACTION_CHANGE_TABLE Literal     88
MM$K_ACTION_DONE_INITS  Literal     88
MM$K_ACTION_DO NOTHING  Literal     88
MM$K_ACTION_DUMMY_3  Literal     88
MM$K_ACTION_DUMMY_4  Literal     88
MM$K_ACTION_DUMMY_5  Literal     88
MM$K_ACTION_DUMMY_6  Literal     88
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_ACTION_DUMMY_7  Literal     88
MM$K_ACTION_DUMMY_8  Literal     88
MM$K_ACTION_INTERNAL_ERROR Literal     88
MM$K_ACTION_LAST_ACTION  Literal     88
MM$K_ACTION_LOAD_AUTOSIZER Literal     88
MM$K_ACTION_MENU_PROMPT  Literal     88
MM$K_ACTION_PRINT_MENU  Literal     88
MM$K_ACTION_PRINT_MENU_HEADING Literal     88
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     88
MM$K_ACTION_START_AUTOSIZER Literal     88
MM$K_STATE_AUTOSIZER_ERROR Literal     88
MM$K_STATE_AUTOSIZER_RUNNING Literal     88
MM$K_STATE_BUILD_DDBS  Literal     88
MM$K_STATE_BUILD_HSTS  Literal     88
MM$K_STATE_CHANGE_TABLE  Literal     88
MM$K_STATE_DONE_INITS  Literal     88
MM$K_STATE_DUMMY_1  Literal     88
MM$K_STATE_DUMMY_2  Literal     88
MM$K_STATE_DUMMY_3  Literal     88
MM$K_STATE_DUMMY_5  Literal     88
MM$K_STATE_DUMMY_7  Literal     88
MM$K_STATE_DUMMY_8  Literal     88
MM$K_STATE_INTERNAL_ERROR Literal     88
MM$K_STATE_LAST_STATE  Literal     88
MM$K_STATE_LOAD_AUTOSIZER Literal     88
MM$K_STATE_MENU_PROMPT  Literal     88
MM$K_STATE_PRINT_MENU  Literal     88
MM$K_STATE_PRINT_MENU_HEADING Literal     88
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     88
MM$K_STATE_START  Literal     88
MM$K_STATE_START_AUTOSIZER Literal     88
MODNAM    Macro    Lib01    93
MOVE_LENGTH   Register   893   916=    918=    918.    920.
NAME    Bind       1104  1109=   1112=   1115=   1118=   1121=   1124-   1127=   1130=   1133=   1136=
            1139=   1142=
NUL2ND_    Macro    Lib04    88      89      90     453     467     470     553     597     599     720
              750     760     782     795     828     846     852    1047    1476    1477
             1483    1484    1489    1542    1583    1599    1602
NUMBER    Bind       904   973a
NUMB_BUF   Local      887   920=    921a    958a    960     973a
NUMB_BUF_PTR   Register   894   920=    921.    958=    960=    960.    962a.   963a.
NUMB_BUF_SZ   Literal    883   887     918
NUMB_SZ    Register   895   921=    960.    973.
PATTERN_BUF   Local     1342  1399=   1402a   1415a   1416.   1429.
PATTERN_BUF_PTR   Local     1340  1398=   1402.   1415.
PATTERN_BUF_SZ   Literal   1329  1342
PATTERN_STRING_SZ  Local     1339  1402=   1428.   1431.
PREPTBL$B_ENTRY_CODE  Field    Lib03  1362a   1363.
PREPTBL$L_ENTRY_TEXT  Field    Lib03   366a   1484.
PREPTBL_INDEX   Local     1333  1358=   1360=   1360.   1362.   1450.
PREPTBL_INIT   Own        340   340=    355.    371=
PTABLE_PTR   Register   673   818-    819.
    Register  1348  1383=   1396aa
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Register  1530  1550=   1552aa  1559.
QUEUE$L_FLINK   Field    Lib03  1195.   1260.
QUEUE_BLKPTR   Register  1183  1193=   1195=   1195a.  1197.
    Register  1249  1258=   1260=   1260a.  1262.
R_CLASS_CODE   RoutForm  1074  1103.
    RoutForm  1149  1188.
    RoutForm  1216  1254.
R_SDN    RoutForm  1149  1189.
SCREEN$K_CLEAR_SCREEN  Literal     90   519    1005
SCREEN$K_COUNT_LINE  Literal     90
SCREEN$K_PRESS_RETURN  Literal     90  1464    1582
SCREEN$K_PRESS_RETURN_MM Literal     90  1491    1604
SCREEN$K_PRESS_RETURN_TM Literal     90
SCREEN$K_TM_MSG   Literal     90   749     781     827
SCREEN_COUNTER   Local      688   722=    746.    751=    755=    755.    778.    783=    787=    787.    824.
             829=    833=    833.
    Local     1332  1352=   1452=   1452.   1453=   1453.   1454=   1454.   1455=   1455.   1456=
             1456.   1457=   1457.   1458=   1458.   1461.   1465=
    Local     1533  1535=   1578.   1584=   1587=   1587.
SDN    Bind       1189  1203.
SELALPHA_INDEX   Local      679   696=    707.    708=    708.
SELALPHA_PTR   Local      683   771=    795.    812=    846.
SELDEVNUMB   Local      684   772=    795.    813=    846.
SEL_ALPHA_PTR   Register   448   459=    467.
SEL_TEXT_PTR   Register   449   460=    467.
SPRTSTAT_PTR   Register  1527  1568=   1570=   1599.
STATUS    Local      512   512=    512.
    Local      999   999=    999.
SUPBLK$B_STRTFRST_CNFGBLK Field    Lib03  1388a
SUPBLK$B_TEST_CLASS  Field    Lib03   809.   1202.   1266.
SUPBLK$T_DEVICE_NAME  Field    Lib03   814a
SUPBLK_PTR   Register   674   807=    809a.   814aa
    Register  1185  1199=   1202a.
    Register  1251  1264=   1266a.
    Register  1349  1385=   1388aa
    Register  1529  1566=
TEXT_PTR   Local      176   206=    208a=   208a.
    Local      336   366=    368a=   368a.
TMENU_OPINP_CHK   Routine    880  1011c
TMTBL$B_ENTRY_ALPHA  Field    Lib03   701a    707=    711=    735a    737.    771a    812a    936a    937.
TMTBL$B_ENTRY_TSTCLA  Field    Lib03   703.    704.    740.    768.    808.    816.   1013.   1030.   1055.
TMTBL_INDEX   Local      682   697=    699=    699.    701.    731=    733=    733.    735.
    Register   899   931=    934=    934.    936.    937.
TQ$K_ACTION_ADVANCE_STATE Literal     88
TQ$K_ACTION_CHANGE_TABLE Literal     88
TQ$K_ACTION_DONE_TEST_QUEUE Literal     88
TQ$K_ACTION_DO_NOTHING  Literal     88
TQ$K_ACTION_DUMMY_3  Literal     88
TQ$K_ACTION_DUMMY_4  Literal     88
TQ$K_ACTION_DUMMY_5  Literal     88
TQ$K_ACTION_GET_DDB  Literal     88
TQ$K_ACTION_INIT_TQ  Literal     88
TQ$K_ACTION_INTERNAL_ERROR Literal     88
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
TQ$K_ACTION_LAST_ACTION  Literal     88
TQ$K_STATE_CHANGE_TABLE  Literal     88
TQ$K_STATE_DONE_TEST_QUEUE Literal     88
TQ$K_STATE_DUMMY_1  Literal     88
TQ$K_STATE_DUMMY_2  Literal     88
TQ$K_STATE_DUMMY_3  Literal     88
TQ$K_STATE_DUMMY_4  Literal     88
TQ$K_STATE_DUMMY_5  Literal     88
TQ$K_STATE_GET_DDB  Literal     88
TQ$K_STATE_INIT_TQ  Literal     88
TQ$K_STATE_INTERNAL_ERROR Literal     88
TQ$K_STATE_LAST_STATE  Literal     88
TRUE    Literal  Lib03   211     371     480     602     945    1433
TSTCLA_NAME_PTR   Local      687   817a    846.
TSTCNFG$B_PREP   Field    Lib03  1410.
TSTCNFG_PTR   Register  1350  1388=   1410a.
TYPE_MAIN_MENU_SELECTIONS Routine    444   503c
TYPE_TEST_MENU_SELECTIONS Routine    670   995c
T_PROMPT_DEFAULT  Bind       106   512a    551.    999a   1045.
VALID_ALPHA_LOWER  Local      434   534=    538.
    Register   898   938=    941.
VALID_ALPHA_STATUS  Register   896   932=    934.    945=    949.
VALID_ALPHA_UPPER  Local      433   533=    534.    537.
    Register   897   937=    938.    940.
VALID_STATUS   Local      432   495=    529.    536=    542.    555.
    Register   660   991=   1021=   1029=   1036.   1049.


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]MENPROMPT.B32;1
    2  Module MENPROMPT
   58  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   59  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   60  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   61  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
 1610  Eludom MENPROMPT
```

```
 KEY TO REFERENCE TYPE FLAGS

      . Fetch
      = Store
      c Routine call
      a Address use
      @ Indirect use
      e External, external routine, or external literal declaration
      f Forward or forward routine declaration
      h Condition handler enabling
      m Map declaration
      u Undeclare declaration


;     Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923         20          2          94          00:00.2
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486        139         28          62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0       1000          00:04.4



;      COMMAND QUALIFIERS

; BLISS MENPROMPT.B32/LIST=MENPROMPT.LIS/CROSS_REFERENCE/DEBUG/OBJECT=MENPROMPT.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  2728 code + 365 data bytes
; Run Time:    01:52.2
; Elapsed Time:    03:24.1
; Lines/CPU Min:     860
; Lexemes/CPU-Min: 70207
; Memory Used: 526 pages
; Compilation Complete
```

```
;    0001   0 %TITLE '*** MENSTATE Module'
;    0002   0 MODULE MENSTATE (    ! The 3 State Tables for Menu Test
;    0003   0    IDENT = 'V01.4'
;    0004   0    ) =
;    0005   0 !++
;    0006   0 ! COPYRIGHT (c) 1980, 1981
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! This module contains the state tables for use by the Menu Test portion of CRD.
;    0032  0 !
;    0033  0 ! Author:
;    0034  0 !
;    0035  0 ! Jack Stansbury
;    0036  0 !
;    0037  0 ! Creation Date:
;    0038  0 !
;    0039  0 ! July 30, 1982
;    0040  0 !
;    0041  0 ! Version:
;    0042  0 !
;    0043  0 ! 6.9
;    0044  0 !
;    0045  0 ! Modified By:
;    0046  0 !
;    0047  0 ! 01-01 Jack Stansbury, March 1, 1982, Version ?.?
;    0048  0 !  Changed the HS state table to allow diagnostics to issue
;    0049  0 !  $DS_Print's from the cleanup section after an abort
;    0050  0 !  (EVRLA prompted this).
;    0051  0 !
;    0052  0 ! 01-02 Jack Stansbury, March 2, 1982, Version ?.?
;    0053  0 !  For related information, see the modification history in CRDSTATE
;    0054  0 !  module (dated March 1, 1983). Essentially, make the Diagnostic_Error
;    0055  0 !  state and the Preparation_Error state more forgiving.
;    0056  0 !
;    0057  0 ! 01-03 Jack Stansbury, March 3, 1983, Version ?.?
;    0058  0 !  Added an Assert statement for the number of typcodes (CRD$K_Numb_Typecodes).
;    0059  0 !  If this number changes from 38, then you must add more rows to these state tables.
;    0060  0 !  Note that these tables (and the ones in CRDSTATE) are VERY dependent on the
;    0061  0 !  value of CRD$K_Numb_Typecodes.
;    0062  0 !
;    0063  0 ! 04 Ron Parker 18-OCT-1984
;    0064  0 !  - Added brief explanation of state table description
;    0065  0 !--
```

```
;    0066  0 %SBTTL 'Libraries'
;    0067  1 BEGIN     ! Begin the MENSTATE module
;    0068  1
;    0069  1 LIBRARY
;    0070  1  '$DS';     ! Use Ds.L32 first
;    0071  1
;    0072  1 LIBRARY
;    0073  1  '$Diag';    ! Use Diag.L32 second
;    0074  1
;    0075  1 LIBRARY
;    0076  1  '$CRD';     ! Use CRD.L32 third
;    0077  1
;    0078  1 LIBRARY
;    0079  1  'Sys$Library:Lib';  ! Use Lib as last resort
```

```
ZZ-ENSAB-2.0    Included Macros and Literals                19-Sep-1985      Fiche 7  Frame C13        Sequence 1394
MENSTATE *** MENSTATE Module    19-Sep-1985 17:20:15 VAX-11 Bliss-32 V4.1-746     Page   4
V01.4  Included Macros and Literals    3-Jan-1985 17:02:35 [DS.CRD.V020]MENSTATE.B32;1   (4)

;    0080  1 %SBTTL 'Included Macros and Literals'
;    0081  1
;    0082  1  $CRD_Literals;    ! Define the CRD literals
```

```
;    0083  1 %SBTTL 'Psect Definitions'
;    0084  1
;    0085  1 PSECT
;    0086  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0087  1
;    0088  1 PSECT
;    0089  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0090  1
;    0091  1 PSECT
;    0092  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0093  1
;    0094  1 PSECT
;    0095  1     GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0096  1 %SBTTL 'Module-Global Static Storage'
;    0097  1
;    0098  1  ModNam ('MENSTATE');  ! Module name for ErrSup macro
;    0099  1
;    0100  1 !+
;    0101  1 !   This is a general description of how the state tables are organized.
;    0102  1 !
;    0103  1 !   The table is a two dimensional array consisting of records.
;    0104  1 !   Each record consists of two words.
;    0105  1 !   The first word is the action to take, in code of course,
;    0106  1 !   and the second word is the next state to enter.
;    0107  1 !
;    0108  1 !   The array is addressed as  TABLE[.Typecode,.State,.Field]  in Bliss.
;    0109  1 !   The .Field term determines which word to look at.
;    0110  1 !-
;    0111  1
```

```
;    0112  1 %SBTTL 'Local Macro Definitions for MM'
;    0113  1
;    0114  1 COMPILETIME
;    0115  1  $MM_Prev_Column$ = 0,  ! The previous column in the Insert_N macro
;    0116  1  $MM_Window_Size$ = 0,  ! The window size in the Insert_N macro
;    0117  1  $MM_Entry$       = 0,  ! Number of entrys put into current row
;    0118  1  $MM_Rows$        = 0;  ! Number of rows put in so far
;    0119  1
;    0120  1 MACRO
; M  0121  1  MM$Fill_N_Entrys (N, New_State, Action_Routine) =
; M  0122  1   REP N OF
; M  0123  1    WORD (%NAME ('MM$K_Action_', Action_Routine),  %NAME ('MM$K_State_', New_State))
; M  0124  1
; M  0125  1   %IF ($MM_Entry$ + N) EQL MM$K_Numb_States
; M  0126  1   %THEN
; M  0127  1    %ASSIGN ($MM_Rows$, $MM_Rows$ + 1)
; M  0128  1    %ASSIGN ($MM_Entry$, 0)
; M  0129  1   %ELSE
; M  0130  1    %IF ($MM_Entry$ + N) GTR MM$K_Numb_States
; M  0131  1    %THEN
; M  0132  1     %ERRORMACRO ('MM$Fill_N_Entrys: Too many entrys (', %NUMBER ($MM_Entry$),
; M  0133  1      ') in row ', %NUMBER ($MM_Rows$))
; M  0134  1    %ELSE
; M  0135  1     %ASSIGN ($MM Entry$, $MM_Entry$ + N)
; M  0136  1    %FI
; M  0137  1   %FI
;    0138  1  %;
;    0139  1
;    0140  1 MACRO
; M  0141  1  MM$Insert_Entrys (New State, Act'on_Routine) [] =
; M  0142  1   MM$Fill_N_Entrys (1, New State, Action Routine)
; M  0143  1
; M  0144  1   %IF NOT %NULL (%REMAINING)
; M  0145  1   %THEN
; M  0146  1    ,MM$Insert_Entrys (%REMAINING)
; M  0147  1   %FI
;    0148  1  %;
;    0149  1
;    0150  1 MACRO
; M  0151  1  MM$Fill_Full_Row =
; M  0152  1   %IF $MM_Entry$ NEQ 0
; M  0153  1   %THEN
; M  0154  1    %ERRORMACRO ('MM$Fill_Full_Row: $MM_Entry$ = ', %NUMBER ($MM_Entry$), ', not zero')
; M  0155  1   %ELSE
; M  0156  1    MM$Fill_N_Entrys (MM$K_Numb_States, Internal_Error, Internal_Error)
; M  0157  1   %FI
;    0158  1  %;
```

```
;    0159  1 MACRO
;    0160  1   !+
;    0161  1   ! Insert entrys into one whole row. The column number tells what column to put 'State' and
;    0162  1   ! 'Action' into. This macro can be called with any number of parameters as long as the
;    0163  1   ! number is 0 mod 3.
;    0164  1   !-
;    0165  1
; M  0166  1   MM$Insert_N (Column, State, Action) [] =
; M  0167  1    %IF %COUNT EQL 0
; M  0168  1    %THEN
; M  0169  1     %ASSIGN ($MM_Prev_Column$, 0)
; M  0170  1    %FI
; M  0171  1
; M  0172  1    %ASSIGN ($MM_Window_Size$, %NAME ('MM$K_State_', Column) - $MM_Prev_Column$)
; M  0173  1
; M  0174  1    %IF $MM_Window_Size$ GTR 0
; M  0175  1    %THEN
; M  0176  1     MM$Fill_N_Entrys ($MM_Window_Size$, Internal_Error, Internal_Error),
; M  0177  1    %FI
; M  0178  1
; M  0179  1    MM$Insert_Entrys (State, Action)
; M  0180  1
; M  0181  1    %ASSIGN ($MM_Prev_Column$, %NAME ('MM$K_State_', Column) + 1)
; M  0182  1
; M  0183  1    %IF NOT %NULL (%REMAINING)
; M  0184  1    %THEN
; M  0185  1     , MM$Insert_N (%REMAINING)
; M  0186  1    %ELSE
; M  0187  1     %ASSIGN ($MM_Window_Size$, MM$K_Numb_States - $MM_Prev_Column$)
; M  0188  1
; M  0189  1     %IF $MM_Window_Size$ GTR 0
; M  0190  1     %THEN
; M  0191  1      , MM$Fill_N_Entrys ($MM_Window_Size$, Internal_Error, Internal_Error)
; M  0192  1     %FI
; M  0193  1    %FI
;    0194  1   %;
```

```
;    0195  1 %SBTTL 'Local Macro Definitions for TQ'
;    0196  1
;    0197  1 COMPILETIME
;    0198  1  $TQ_Prev_Column$ = 0,  ! The previous column in the Insert_N macro
;    0199  1  $TQ_Window_Size$ = 0,  ! The window size in the Insert_N macro
;    0200  1  $TQ_Entry$       = 0,  ! Number of entrys put into current row
;    0201  1  $TQ_Rows$        = 0;  ! Number of rows put in so far
;    0202  1
;    0203  1 MACRO
; M  0204  1  TQ$Fill_N_Entrys (N, New_State, Action_Routine) =
; M  0205  1   REP N OF
; M  0206  1    WORD (%NAME ('TQ$K_Action_', Action_Routine),  %NAME ('TQ$K_State_', New_State))
; M  0207  1
; M  0208  1   %IF ($TQ_Entry$ + N) EQL TQ$K_Numb_States
; M  0209  1   %THEN
; M  0210  1    %ASSIGN ($TQ_Rows$, $TQ_Rows$ + 1)
; M  0211  1    %ASSIGN ($TQ_Entry$, 0)
; M  0212  1   %ELSE
; M  0213  1    %IF ($TQ_Entry$ + N) GTR TQ$K Numb_States
; M  0214  1    %THEN
; M  0215  1     %ERRORMACRO ('TQ$Fill_N_Entrys: Too many entrys (', %NUMBER ($TQ_Entry$),
; M  0216  1      ') in row ', %NUMBER ($TQ_Rows$))
; M  0217  1    %ELSE
; M  0218  1     %ASSIGN ($TQ_Entry$, $TQ_Entry$ + N)
; M  0219  1    %FI
; M  0220  1   %FI
;    0221  1  %;
;    0222  1
;    0223  1 MACRO
; M  0224  1  TQ$Insert_Entrys (New_State, Action Routine) [] -
; M  0225  1   TQ$Fill_N_Entrys (1, New_State, Action Routine)
; M  0226  1
; M  0227  1   %IF NOT %NULL (%REMAINING)
; M  0228  1   %THEN
; M  0229  1    ,TQ$Insert_Entrys (%REMAINING)
; M  0230  1   %FI
;    0231  1  %;
;    0232  1
;    0233  1 MACRO
; M  0234  1  TQ$Fill_Full_Row =
; M  0235  1   %IF $TQ_Entry$ NEQ 0
; M  0236  1   %THEN
; M  0237  1    %ERRORMACRO ('TQ$Fill Full Row: $TQ_Entry$ = ', %NUMBER ($TQ_Entry$), ', not zero')
; M  0238  1   %ELSE
; M  0239  1    TQ$Fill N Entrys (TQ$K_Numb States, Internal_Error, Internal_Error)
; M  0240  1   %FI
;    0241  1  %;
```

```
;    0242  1 MACRO
;    0243  1  !+
;    0244  1  ! Insert entrys into one whole row. The column number tells what column to put 'State' and
;    0245  1  ! 'Action' into. This macro cann be called with any number of parameters as long as the
;    0246  1  ! number is mod 3.
;    0247  1  !-
;    0248  1
; M  0249  1  TQ$Insert_N (Column, State, Action) [] =
; M  0250  1   %IF %COUNT EQL 0
; M  0251  1   %THEN
; M  0252  1    %ASSIGN ($TQ_Prev_Column$, 0)
; M  0.53  1   %FI
; M  0254  1
; M  0255  1   %ASSIGN ($TQ_Window_Size$, %NAME ('TQ$K_State_', Column) - $TQ_Prev_Column$)
; M  0256  1
; M  0257  1   %IF $TQ_Window_Size$ GTR 0
; M  0258  1   %THEN
; M  0259  1    TQ$Fill_N_Entrys ($TQ_Window_Size$, Internal_Error, Internal_Error),
; M  0260  1   %FI
; M  0261  1
; M  0262  1   TQ$Insert_Entrys (State, Action)
; M  0263  1
; M  0264  1   %ASSIGN ($TQ_Prev_Column$, %NAME ('TQ$K_State_', Column) + 1)
; M  0265  1
; M  0266  1   %IF NOT %NULL (%REMAINING)
; M  0267  1   %THEN
; M  0268  1    , TQ$Insert_N (%REMAINING)
; M  0269  1   %ELSE
; M  0270  1    %ASSIGN ($TQ_Window_Size$, TQ$K_Numb_States - $TQ_Prev_Column$)
; M  0271  1
; M  0272  1    %IF $TQ_Window_Size$ GTR 0
; M  0273  1    %THEN
; M  0274  1     , TQ$Fill_N_Entrys ($TQ_Window_Size$, Internal_Error, Internal_Error)
; M  0275  1    %FI
; M  0276  1   %FI
;    0277  1  %;
```

```
;    0278  1 %SBTTL 'Local Macro Definitions for HS'
;    0279  1
;    0280  1 COMPILETIME
;    0281  1  $HS_Prev_Column$ = 0,  ! The previous column in the Insert_N macro
;    0282  1  $HS_Window_Size$ = 0,  ! The window size in the Insert_N macro
;    0283  1  $HS_Entry$       = 0,  ! Number of entrys put into current row
;    0284  1  $HS_Rows$        = 0;  ! Number of rows put in so far
;    0285  1
;    0286  1 MACRO
; M  0287  1  HS$Fill_N_Entrys (N, New_State, Action_Routine) =
; M  0288  1   REP N OF
; M  0289  1    WORD (%NAME ('HS$K_Action_', Action_Routine),  %NAME ('HS$K_State_', New_State))
; M  0290  1
; M  0291  1   %IF ($HS_Entry$ + N) EQL HS$K_Numb_States
; M  0292  1   %THEN
; M  0293  1    %ASSIGN ($HS_Rows$, $HS_Rows$ + 1)
; M  0294  1    %ASSIGN ($HS_Entry$, 0)
; M  0295  1   %ELSE
; M  0296  1    %IF ($HS_Entry$ + N) GTR HS$K_Numb States
; M  0297  1    %THEN
; M  0298  1     %ERRORMACRO ('HS$Fill_N_Entrys: Too many entrys (', %NUMBER ($HS_Entry$),
; M  0299  1      ') in row ', %NUMBER ($HS_Rows$))
; M  0300  1    %ELSE
; M  0301  1     %ASSIGN ($HS Entry$, $HS_Entry$ + N)
; M  0302  1    %FI
; M  0303  1   %FI
;    0304  1  %;
;    0305  1
;    0306  1 MACRO
; M  0307  1  HS$Insert Entrys (New State, Action_Routine) [] =
; M  0308  1   HS$Fill_N_Entrys (1, New_State, Action_Routine)
; M  0309  1
; M  0310  1   %IF NOT %NULL (%REMAINING)
; M  0311  1   %THEN
; M  0312  1    ,HS$Insert_Entrys (%REMAINING)
; M  0313  1   %FI
;    0314  1  %;
;    0315  1
;    0316  1 MACRO
; M  0317  1  HS$Fill_Full_Row =
; M  0318  1   %IF $HS_Entry$ NEQ 0
; M  0319  1   %THEN
; M  0320  1    %ERRORMACRO ('HS$Fill_Full_Row: $HS_Entry$ = ', %NUMBER ($HS_Entry$), ', not zero')
; M  0321  1   %ELSE
; M  0322  1    HS$Fill_N_Entrys (HS$K_Numb_States, Internal_Error, Internal_Error)
; M  0323  1   %FI
;    0324  1  %;
```

```
;    0325  1 MACRO
;    0326  1  !+
;    0327  1  ! Insert entrys into one whole row. The column number te!'s what column to put 'State' and
;    0328  1  ! 'Action' into. This macro cann be called with any numbe! of parameters as long as the
;    0329  1  ! number is mod 3.
;    0330  1  !-
;    0331  1
; M  0332  1  HS$Insert_N (Column, State, Action) [] =
; M  0333  1   %IF %COUNT EQL 0
; M  0334  1   %THEN
; M  0335  1    %ASSIGN ($HS_Prev_Column$, 0)
; M  0336  1   %FI
; M  0337  1
; M  0338  1   %ASSIGN ($HS_Window_Size$, %NAME ('HS$K_State_', Column) - $HS_Prev_Co umn$
; M  0339  1
; M  0340  1   %IF $HS_Window_Size$ GTR 0
; M  0341  1   %THEN
; M  0342  1    HS$Fill_N_Entrys ($HS_Window_Size$, Internal_Error, Internal Error),
; M  0343  1   %FI
; M  0344  1
; M  0345  1   HS$Insert_Entrys (State, Action)
; M  0346  1
; M  0347  1   %ASSIGN ($HS_Prev_Column$, %NAME ('HS$K_State_', Column) + 1)
; M  0348  1
; M  0349  1   %IF NOT %NULL (%REMAINING)
; M  0350  1   %THEN
; M  0351  1    , HS$Insert_N (%REMAINING)
; M  0352  1   %ELSE
; M  0353  1    %ASSIGN ($HS_Window_Size$, HS$K_Numb_States - $HS_Prev_Column$)
; M  0354  1
; M  0355  1    %IF $HS_Window_Size$ GTR 0
; M  0356  1    %THEN
; M  0357  1     , HS$Fill_N_Entrys ($HS_Window_Size$, Internal_Error, Internal_Error)
; M  0358  1    %FI
; M  0359  1   %FI
;    0360  1  %;
```

```
;    0361  1 %SBTTL 'GLOBAL OWN Storage'
;    0362  1
;    0363  1 GLOBAL
;    0364  1  CRD$GL_Previous_State,  ! The previous state
;    0365  1
;    0366  1  CRD$GL_MM_Current_State, ! The current state for the Main Menu
;    0367  1  CRD$GL_TQ_Current_State, ! The current state for the Test Queue Table
;    0368  1  CRD$GL_HS_Current_State, ! The current state for the Hardware Support
;    0369  1
;    0370  1  CRD$GL_TQ_Current_TQ_Entry, ! An index into the Test Queue Table, for the current entry
;    0371  1
;    0372  1  CRD$GA_HS_Current_DDB_Ptr, ! The current DDB that is being tested
;    0373  1  CRD$GA_HS_Current_HSB_Ptr, ! The current Hardware Support Block being tested
;    0374  1  CRD$GL_HS_Current_HSB_Numb, ! The number of the current Hardware Support Block
;    0375  1
;    0376  1  CRD$GB_Current_State_Table : BYTE;
;    0377  1      ! 0 => Using the MM state table
;    0378  1      ! 1 => Using the TQ state table
;    0379  1      ! 2 => Using the HS state table
```

```
;   0380  1 %SBTTL 'Main Menu Global Bindings'
;   0381  1
; P 0382  1 CRD_Assert ((CRD$K_Numb_Typecodes EQL 38),          ![03]
; L 0383  1  'You must make additions to these tables since the number of typcodes has changed');   ![03]
; %PRINT: ASSERT: The assertion is true.
;   0384  1
;   0385  1 GLOBAL BIND
;   0386  1  CRD$GAW_MM_State_Table =
;   0387  1   UPLIT (
; P 0388  1    MM$Insert_N (
; P 0389  1      Load_AutoSizer,  AutoSizer_Error,  AutoSizer_Error,
; P 0390  1      Set_Flags_AutoSizer,  AutoSizer_Error,  AutoSizer_Error,
; P 0391  1      Start_AutoSizer,  AutoSizer_Error,  AutoSizer_Error,
; P 0392  1      AutoSizer_Running,  AutoSizer_Running,  Do_Nothing
;   0393  1         ),
;   0394  1      ! This row is for the General typecode (Row 0)
;   0395  1
; P 0396  1    MM$Insert_N (
; P 0397  1      !+
; P 0398  1      ! Current State  Next State  Current Action Routine
; P 0399  1      !-
; P 0400  1      Start,   Print_Menu_Heading, Advance_State,
; P 0401  1      Print_Menu_Heading, Load_AutoSizer,  Print_Menu_Heading,
; P 0402  1      Load_AutoSizer,  Set_Flags_AutoSizer, Load_AutoSizer,
; P 0403  1      Set_Flags_AutoSizer,  Start_AutoSizer, Set_Flags_AutoSizer,
; P 0404  1      Start_AutoSizer,  AutoSizer_Running, Start_AutoSizer,
; P 0405  1      AutoSizer_Running,  Build_DDBs,  Advance_State,
; P 0406  1      Build_DDBs,  Build_HSTs,  Build_DDBs,
; P 0407  1      Build_HSTs,  Done_Inits,  Build_HSTs,
; P 0408  1      Done_Inits,  Print_Menu,  Done_Inits,
; P 0409  1      Print_Menu,  Change_Table,  Print_Menu,
; P 0410  1      Change_Table,  Print_Menu,  Change_Table
;   0411  1         ),
;   0412  1      ! This row is for the DS_Prompt typecode (Row 1)
;   0413  1
; P 0414  1    MM$Insert_N (
; P 0415  1      AutoSizer_Running, AutoSizer_Error, AutoSizer_Error,
; P 0416  1      Change_Table,  Change_Table,  Menu_Prompt
;   0417  1         ),
;   0418  1      ! This row is for the User_Prompt typecode (Row 2)
;   0419  1
; P 0420  1    MM$Insert_N (
; P 0421  1      Load_AutoSizer,  AutoSizer_Error, AutoSizer_Error,
; P 0422  1      Set_Flags_AutoSizer,  AutoSizer_Error, AutoSizer_Error,
; P 0423  1      Start_AutoSizer,  AutoSizer_Error, AutoSizer_Error,
; P 0424  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0425  1         ),
;   0426  1      ! This row is for the General_Error typecode (Row 3)
;   0427  1
; P 0428  1    MM$Insert N (
; P 0429  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0430  1         ),
;   0431  1      ! This row is for the ErrSup typecode (Row 4)
;   0432  1
; P 0433  1    MM$Insert N (
```

```
; P 0434  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0435  1          ),
;   0436  1      ! This row is for the ErrSys typecode (Row 5)
;   0437  1
; P 0438  1    MM$Insert_N (
; P 0439  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0440  1          ),
;   0441  1      ! This row is for the ErrHard typecode (Row 6)
;   0442  1
; P 0443  1    MM$Insert_N (
; P 0444  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0445  1          ),
;   0446  1      ! This row is for the ErrSoft typecode (Row 7)
;   0447  1
; P 0448  1    MM$Insert_N (
; P 0449  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0450  1          ),
;   0451  1      ! This row is for the ErrDev typecode (Row 8)
;   0452  1
; P 0453  1    MM$Insert_N (
; P 0454  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0455  1          ),
;   0456  1      ! This row is for the Error_Body typecode (Row 9)
;   0457  1
; P 0458  1    MM$Insert_N (
; P 0459  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0460  1          ),
;   0461  1      ! This row is for the Error_End typecode (Row 10)
;   0462  1
; P 0463  1    MM$Insert_N (
; P 0464  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0465  1          ),
;   0466  1      ! This row is for the Exception_Head typecode (Row 11)
;   0467  1
; P 0468  1    MM$Insert_N (
; P 0469  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0470  1          ),
;   0471  1      ! This row is for the Exception typecode (Row 12)
;   0472  1
; P 0473  1    MM$Insert_N (
; P 0474  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0475  1          ),
;   0476  1      ! This row is for the Err_Halt typecode (Row 13)
;   0477  1
; P 0478  1    MM$Insert_N (
; P 0479  1      AutoSizer_Running,  AutoSizer_Running, Do_Nothing
;   0480  1          ),
;   0481  1      ! This row is for the Summary typecode (Row 14)
;   0482  1
; P 0483  1    MM$Insert_N (
; P 0484  1      AutoSizer_Running, AutoSizer_Running, Do_Nothing
;   0485  1          ),
;   0486  1      ! This row is for the Program_Start typecode (Row 15)
;   0487  1
; P 0488  1    MM$Insert N (
```

```
;P 0489  1      AutoSizer_Running,  AutoSizer_Running, Do_Nothing
;  0490  1          ),
;  0491  1      ! This row is for the Program_End typecode (Row 16)
;  0492  1
;P 0493  1    MM$Insert_N (
;P 0494  1      AutoSizer_Running,  AutoSizer_Running, Do_Nothing
;  0495  1          ),
;  0496  1      ! This row is for the First_Pass typecode (Row 17)
;  0497  1
;P 0498  1    MM$Insert_N (
;P 0499  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;  0500  1          ),
;  0501  1      ! This row is for the No_Tests typecode (Row 18)
;  0502  1
;P 0503  1    MM$Insert_N (
;P 0504  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;  0505  1          ),
;  0506  1      ! This row is for the Abort_Test typecode (Row 19)
;  0507  1
;P 0508  1    MM$Insert_N (
;P 0509  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;  0510  1          ),
;  0511  1      ! This row is for the Abort_Program typecode (Row 20)
;  0512  1
;P 0513  1    MM$Insert_N (
;P 0514  1      Load_AutoSizer,  AutoSizer_Error, AutoSizer_Error,
;P 0515  1      Set Flags_AutoSizer,  AutoSizer_Error, AutoSizer_Error,
;P 0516  1      Start_AutoSizer,  AutoSizer_Error, AutoSizer_Error,
;P 0517  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;  0518  1          ),
;  0519  1      ! This row is for the Command_Err typecode (Row 21)
;  0520  1
;P 0521  1    MM$Insert_N (
;P 0522  1      Load_AutoSizer,  AutoSizer_Error, AutoSizer_Error,
;P 0523  1      Set Flags_AutoSizer,  AutoSizer_Error, AutoSizer_Error,
;P 0524  1      Start_AutoSizer,  AutoSizer_Error, AutoSizer_Error,
;P 0525  1      AutoSizer Running,  AutoSizer_Error, AutoSizer_Error
;  0526  1          ),
;  0527  1      ! This row is for the Command_Out typecode (Row 22)
;  0528  1
;P 0529  1    MM$Insert_N (
;P 0530  1      AutoSizer Running,  AutoSizer_Running, Do_Nothing
;  0531  1          ),
;  0532  1      ! This row is for the Program_Info typecode (Row 23)
;  0533  1
;P 0534  1    MM$Insert_N (
;P 0535  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;  0536  1          ),
;  0537  1      ! This row is for the Start_Err typecode (Row 24)
;  0538  1
;P 0539  1    MM$Insert_N (
;P 0540  1      AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;  0541  1          ),
;  0542  1      ! This row is for the Sequence_Error typecode (Row 25)
;  0543  1
```

```
;   0544  1     MM$Fill_Full_Row,
;   0545  1       ! This row is for the CRD_AutoTest typecode (Row 26)
;   0546  1
; P 0547  1     MM$Insert_N (
; P 0548  1       AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0549  1           ),
;   0550  1       ! This row is for the ErrPrep typecode (Row 27)
;   0551  1
; P 0552  1     MM$Insert_N (
; P 0553  1       AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0554  1           ),
;   0555  1       ! This row is for the Param_Error typecode (Row 28)
;   0556  1
; P 0557  1     MM$Insert N (
; P 0558  1       Start,   Print_Menu_Heading, Do_Nothing
;   0559  1           ),
;   0560  1       ! This row is for the DS_Start typecode (Row 29)
;   0561  1
;   0562  1     MM$Fill_Full_Row,
;   0563  1       ! This row is for the Script_Pnf typecode (Row 30)
;   0564  1
;   0565  1     MM$Fill_Full_Row,
;   0566  1       ! This row is for the Script_Skip typecode (Row 31)
;   0567  1
; P 0568  1     MM$Insert_N (
; P 0569  1       Change_Table,  Change_Table,  Menu_Prompt
;   0570  1           ),
;   0571  1       ! This row is for the Script_Prompt typecode (Row 32)
;   0572  1
;   0573  1     MM$Fill_Full_Row,
;   0574  1       ! This row is for the Script_Echo typecode (Row 33)
;   0575  1
; P 0576  1     MM$Insert_N (
; P 0577  1       AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0578  1           ),
;   0579  1       ! This row is for the QIO_NoDriver typecode (Row 34)
;   0580  1
; P 0581  1     MM$Insert_N (
; P 0582  1       AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0583  1           ),
;   0584  1       ! This row is for the QIO_WrongVer typecode (Row 35)
;   0585  1
; P 0586  1     MM$Insert_N (
; P 0587  1       AutoSizer_Running,  AutoSizer_Error, AutoSizer_Error
;   0588  1           ),
;   0589  1       ! This row is for the QIO_InvAdp typecode (Row 36)
;   0590  1
;   0591  1     MM$Fill_Full_Row
;   0592  1       ! This row is for the Start_List typecode (Row 37)
;   0593  1
;   0594  1   ) : MM_State Table_Structure [CRD$K_Numb_TypeCodes, MM$K_Numb_States];
```

```
;    0595  1 %SBTTL 'Test Queue Global Bindings'
;    0596  1
;    0597  1 GLOBAL BIND
;    0598  1  CRD$GAW_TQ_State_Table =
;    0599  1   UPLIT (
;    0600  1    TQ$Fill_Full_Row,
;    0601  1      ! This row is for the General typecode (Row 0)
;    0602  1
; P  0603  1    TQ$Insert_N (
; P  0604  1      !+
; P  0605  1      ! Current State  Next State  Current Action Routine
; P  0606  1      !-
; P  0607  1
; P  0608  1      Init_TQ,  Get_DDB,  Init_TQ,
; P  0609  1      Get_DDB,  Change_Table,  Get_DDB,
; P  0610  1      Change_Table,  Get_DDB,  Change_Table,
; P  0611  1      Done_Test_Queue, Done_Test_Queue, Done_Test_Queue
;    0612  1         ),
;    0613  1      ! This row is for the DS_Prompt typecode (Row 1)
;    0614  1
;    0615  1    TQ$Fill_Full_Row,
;    0616  1      ! This row is for the User_Prompt typecode (Row 2)
;    0617  1
;    0618  1    TQ$Fill_Full_Row,
;    0619  1      ! This row is for the General_Error typecode (Row 3)
;    0620  1
;    0621  1    TQ$Fill_Full_Row,
;    0622  1      ! This row is for the ErrSup typecode (Row 4)
;    0623  1
;    0624  1    TQ$Fill_Full_Row,
;    0625  1      ! This row is for the ErrSys typecode (Row 5)
;    0626  1
;    0627  1    TQ$Fill_Full_Row,
;    0628  1      ! This row is for the ErrHard typecode (Row 6)
;    0629  1
;    0630  1    TQ$Fill_Full_Row,
;    0631  1      ! This row is for the ErrSoft typecode (Row 7)
;    0632  1
;    0633  1    TQ$Fill_Full_Row,
;    0634  1      ! This row is for the ErrDev typecode (Row 8)
;    0635  1
;    0636  1    TQ$Fill_Full_Row,
;    0637  1      ! This row is for the Error_Body typecode (Row 9)
;    0638  1
;    0639  1    TQ$Fill_Full_Row,
;    0640  1      ! This row is for the Error_End typecode (Row 10)
;    0641  1
;    0642  1    TQ$Fill_Full_Row,
;    0643  1      ! This row is for the Exception_Head typecode (Row 11)
;    0644  1
;    0645  1    TQ$Fill_Full_Row,
;    0646  1      ! This row is for the Exception typecode (Row 12)
;    0647  1
;    0648  1    TQ$Fill Full_Row,
;    0649  1      ! This row is for the Err_Halt typecode (Row 13)
```

```
;    0650  1
;    0651  1     TQ$Fill_Full_Row,
;    0652  1       ! This row is for the Summary typecode (Row 14)
;    0653  1
;    0654  1     TQ$Fill_Full_Row,
;    0655  1       ! This row is for the Program_Start typecode (Row 15)
;    0656  1
;    0657  1     TQ$Fill_Full_Row,
;    0658  1       ! This row is for the Program_End typecode (Row 16)
;    0659  1
;    0660  1     TQ$Fill_Full_Row,
;    0661  1       ! This row is for the First_Pass typecode (Row 17)
;    0662  1
;    0663  1     TQ$Fill_Full_Row,
;    0664  1       ! This row is for the No_Tests typecode (Row 18)
;    0665  1
;    0666  1     TQ$Fill_Full_Row,
;    0667  1       ! This row is for the Abort_Test typecode (Row 19)
;    0668  1
;    0669  1     TQ$Fill_Full_Row,
;    0670  1       ! This row is for the Abort_Program typecode (Row 20)
;    0671  1
;    0672  1     TQ$Fill_Full_Row,
;    0673  1       ! This row is for the Command_Err typecode (Row 21)
;    0674  1
;    0675  1     TQ$Fill_Full_Row,
;    0676  1       ! This row is for the Command_Out typecode (Row 22)
;    0677  1
;    0678  1     TQ$Fill_Full_Row,
;    0679  1       ! This row is for the Program_Info typecode (Row 23)
;    0680  1
;    0681  1     TQ$Fill_Full_Row,
;    0682  1       ! This row is for the Start_Err typecode (Row 24)
;    0683  1
;    0684  1     TQ$Fill_Full_Row,
;    0685  1       ! This row is for the Sequence_Error typecode (Row 25)
;    0686  1
;    0687  1     TQ$Fill_Full_Row,
;    0688  1       ! This row is for the CRD_AutoTest typecode (Row 26)
;    0689  1
;    0690  1     TQ$Fill_Full_Row,
;    0691  1       ! This row is for the ErrPrep typecode (Row 27)
;    0692  1
;    0693  1     TQ$Fill_Full_Row,
;    0694  1       ! This row is for the Param_Error typecode (Row 28)
;    0695  1
;    0696  1     TQ$Fill_Full_Row,
;    0697  1       ! This row is for the DS_Start typecode (Row 29)
;    0698  1
;    0699  1     TQ$Fill_Full_Row,
;    0700  1       ! This row is for the Script_Pnf typecode (Row 30)
;    0701  1
;    0702  1     TQ$Fill_Full_Row,
;    0703  1       ! This row is for the Script_Skip typecode (Row 31)
;    0704  1
```

```
;    0705  1     TQ$Fill_Full_Row,
;    0706  1       ! This row is for the Script_Prompt typecode (Row 32)
;    0707  1
;    0708  1     TQ$Fill_Full_Row,
;    0709  1       ! This row is for the Script_Echo typecode (Row 33)
;    0710  1
;    0711  1     TQ$Fill_Full_Row,
;    0712  1       ! This row is for the QIO_NoDriver typecode (Row 34)
;    0713  1
;    0714  1     TQ$Fill_Full_Row,
;    0715  1       ! This row is for the QIO_WrongVer typecode (Row 35)
;    0716  1
;    0717  1     TQ$Fill_Full_Row,
;    0718  1       ! This row is for the QIO_InvAdp typecode (Row 36)
;    0719  1
;    0720  1     TQ$Fill_Full Row
;    0721  1       ! This row is for the Start_List typecode (Row 37)
;    0722  1
;    0723  1    ) : TQ_State_Table_Structure [CRD$K_Numb_TypeCodes, TQ$K_Numb_States];
```

```
;    0724  1 %SBTTL 'Hardware Support Global Bindings'
;    0725  1
;    0726  1 GLOBAL BIND
;    0727  1  CRD$GAW_HS_State_Table =
;    0728  1   UPLIT (
; P  0729  1    HS$Insert_N (
; P  0730  1      !+
; P  0731  1      ! Current State  Next State  Current Action Routine
; P  0732  1      !-
; P  0733  1
; P  0734  1      Diagnostic_Running, Diagnostic_Running, Do_Nothing,
; P  0735  1      Load_Error,  Load_Error,  Do_Nothing,
; P  0736  1      Start_Error,  Start_Error,  Do_Nothing,
; P  0737  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P  0738  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [01]
; P  0739  1      Internal_Error,  Internal_Error,  Internal_Error
;    0740  1         ),
;    0741  1      ! This row is for the General typecode (Row 0)
;    0742  1
; P  0743  1    HS$Insert_N (
; P  0744  1      Init_HS,  Advance_Diagnostic, Init_HS,
; P  0745  1      Advance_Diagnostic, Load_General_Com, Advance_Diagnostic,
; P  0746  1      Load_General_Com, Advance_General_Com, Load_General_Com,
; P  0747  1      Advance_General_Com, Load_General_Com, Advance_General_Com,
; P  0748  1      Done_General_Coms, Load_Load_Com,  Done_General_Coms,
; P  0749  1      Load_Load_Com,  Load_Set_Flags_Com, Load_Load_Com,
; P  0750  1      Load_Set_Flags_Com, Load_Set_Event_Com, Load_Set_Flags_Com,
; P  0751  1      Load_Set_Event_Com, Load_Select_Com, Load_Set_Event_Com,
; P  0752  1      Load_Select_Com, Load Start_Com,  Load_Select_Com,
; P  0753  1      Load_Start_Com,  Diagnostic_Running, Load_Start_Com,
; P  0754  1      Diagnostic_Running, Advance_Diagnostic, Advance_State,
; P  0755  1      Change_Table,  Change_Table,  Change_Table,
; P  0756  1      Load_Error,  Advance_Diagnostic, Advance_State,
; P  0757  1      Start_Error,  Advance_Diagnostic, Advance_State,
; P  0758  1      Diagnostic_Error, Change_Table,  Advance_State,
; P  0759  1      Preparation_Error, Advance_Diagnostic, Advance_State,
; P  0760  1      Internal_Error,  Internal_Error,  Internal_Error
;    0761  1         ),
;    0762  1      ! This row is for the DS_Prompt typecode (Row 1)
;    0763  1
; P  0764  1    HS$Insert_N (
; P  0765  1      Diagnostic_Running, Diagnostic_Error, Diagnostic_Error,
; P  0766  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P  0767  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P  0768  1      Internal_Error,  Internal_Error,  Internal_Error
;    0769  1         ),
;    0770  1      ! This row is for the User_Prompt typecode (Row 2)
;    0771  1
; P  0772  1    HS$Insert_N (
; P  0773  1      Load_Set_Flags_Com,  Load_Error,  Load_Error,
; P  0774  1      Load_Set_Event_Com,  Start_Error,  Start_Error,
; P  0775  1      Load_Select_Com,  Start_Error,  Start_Error,
; P  0776  1      Load_Start_Com,  Start_Error,  Start_Error,
; P  0777  1      Diagnostic_Running,  Start_Error,  Start_Error,
; P  0778  1      Load_Error,  Load_Error,  Do_Nothing,
```

```
;P 0779  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0780  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0781  1      Internal_Error,  Internal_Error,  Internal_Error
;   0782  1        ),
;   0783  1      ! This row is for the General_Error typecode (Row 3)
;   0784  1
;P 0785  1    HS$Insert_N (
;P 0786  1      Internal_Error,  Internal_Error,  Internal_Error
;   0787  1        ),
;   0788  1      ! This row is for the ErrSup typecode (Row 4)
;   0789  1
;P 0790  1    HS$Insert_N (
;P 0791  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
;P 0792  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,  ! [02]
;P 0793  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0794  1      Internal_Error,  Internal_Error,  Internal_Error
;   0795  1        ),
;   0796  1      ! This row is for the ErrSys typecode (Row 5)
;   0797  1
;P 0798  1    HS$Insert_N (
;P 0799  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
;P 0800  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,  ! [02]
;P 0801  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0802  1      Internal_Error,  Internal_Error,  Internal_Error
;   0803  1        ),
;   0804  1      ! This row is for the ErrHard typecode (Row 6)
;   0805  1
;P 0806  1    HS$Insert_N (
;P 0807  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
;P 0808  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,  ! [02]
;P 0809  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0810  1      Internal_Error,  Internal_Error,  Internal_Error
;   0811  1        ),
;   0812  1      ! This row is for the ErrSoft typecode (Row 7)
;   0813  1
;P 0814  1    HS$Insert_N (
;P 0815  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
;P 0816  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,  ! [02]
;P 0817  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0818  1      Internal_Error,  Internal_Error,  Internal_Error
;   0819  1        ),
;   0820  1      ! This row is for the ErrDev typecode (Row 8)
;   0821  1
;P 0822  1    HS$Insert_N (
;P 0823  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
;P 0824  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0825  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0826  1      Internal_Error,  Internal_Error,  Internal_Error
;   0827  1        ),
;   0828  1      ! This row is for the Error_Body typecode (Row 9)
;   0829  1
;P 0830  1    HS$Insert_N (
;P 0831  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
;P 0832  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0833  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
```

```
;P 0834  1      Internal_Error,  Internal_Error,  Internal_Error
;  0835  1          ),
;  0836  1      ! This row is for the Error_End typecode (Row 10)
;  0837  1
;P 0838  1    HS$Insert_N (
;P 0839  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
;P 0840  1      Internal_Error,  Internal_Error,  Internal_Error
;  0841  1          ),
;  0842  1      ! This row is for the Exception_Head typecode (Row 11)
;  0843  1
;P 0844  1    HS$Insert_N (
;P 0845  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
;P 0846  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0847  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0848  1      Internal_Error,  Internal_Error,  Internal_Error
;  0849  1          ),
;  0850  1      ! This row is for the Exception typecode (Row 12)
;  0851  1
;P 0852  1    HS$Insert_N (
;P 0853  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
;P 0854  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0855  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0856  1      Internal_Error,  Internal_Error,  Internal_Error
;  0857  1          ),
;  0858  1      ! This row is for the Err_Halt typecode (Row 13)
;  0859  1
;P 0860  1    HS$Insert_N (
;P 0861  1      Diagnostic_Running,  Diagnostic_Running, Do_Nothing,
;P 0862  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,  ! [02]
;P 0863  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0864  1      Internal_Error,  Internal_Error,  Internal_Error
;  0865  1          ),
;  0866  1      ! This row is for the Summary typecode (Row 14)
;  0867  1
;P 0868  1    HS$Insert_N (
;P 0869  1      Diagnostic_Running,  Diagnostic_Running, Do_Nothing,
;P 0870  1      Internal_Error,  Internal_Error,  Internal_Error
;  0871  1          ),
;  0872  1      ! This row is for the Program_Start typecode (Row 15)
;  0873  1
;P 0874  1    HS$Insert_N (
;P 0875  1      Diagnostic_Running,  Diagnostic_Running, Do_Nothing,
;P 0876  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,  ! [02]
;P 0877  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0878  1      Internal_Error,  Internal_Error,  Internal_Error
;  0879  1          ),
;  0880  1      ! This row is for the Program_End typecode (Row 16)
;  0881  1
;P 0882  1    HS$Insert_N (
;P 0883  1      Diagnostic_Running,  Diagnostic_Running, Do_Nothing,
;P 0884  1      Internal_Error,  Internal_Error,  Internal_Error
;  0885  1          ),
;  0886  1      ! This row is for the First_Pass typecode (Row 17)
;  0887  1
;P 0888  1    HS$Insert_N (
```

```
;P 0889  1       Diagnostic_Running, Diagnostic_Error, Diagnostic_Error,
;P 0890  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0891  1       Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0892  1       Internal_Error,  Internal_Error,  Internal_Error
;   0893  1          ),
;   0894  1       ! This row is for the No_Tests typecode (Row 18)
;   0895  1
;P 0896  1     HS$Insert_N (
;P 0897  1       Diagnostic_Running, Diagnostic_Error, Diagnostic_Error,
;P 0898  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0899  1       Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0900  1       Internal_Error,  Internal_Error,  Internal_Error
;   0901  1          ),
;   0902  1       ! This row is for the Abort_Test typecode (Row 19)
;   0903  1
;P 0904  1     HS$Insert_N (
;P 0905  1       Diagnostic_Running, Diagnostic_Error, Diagnostic_Error,
;P 0906  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0907  1       Preparation_Error, Preparation_Error, Do_Nothing,
;P 0908  1       Internal_Error,  Internal_Error,  Internal_Error
;   0909  1          ),
;   0910  1       ! This row is for the Abort_Program typecode (Row 20)
;   0911  1
;P 0912  1     HS$Insert_N (
;P 0913  1       Load_Set_Flags_Com,  Load_Error,  Load_Error,
;P 0914  1       Load_Set_Event_Com,  Start_Error,  Start_Error,
;P 0915  1       Load_Select_Com,  Start_Error,  Start_Error,
;P 0916  1       Load_Start_Com,  Start_Error,  Start_Error,
;P 0917  1       Diagnostic_Running, Start_Error,  Start_Error,
;P 0918  1       Load_Error,  Load_Error,  Do_Nothing,
;P 0919  1       Start_Error,  Start_Error,  Do_Nothing,
;P 0920  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0921  1       Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
;P 0922  1       Internal_Error,  Internal_Error,  Internal_Error
;   0923  1          ),
;   0924  1       ! This row is for the Command_Err typecode (Row 21)
;   0925  1
;P 0926  1     HS$Insert_N (
;P 0927  1       Load_Set_Flags_Com,  Load_Set_Flags_Com, Do_Nothing,
;P 0928  1       Load_Set_Event_Com,  Load_Set_Event_Com, Do_Nothing,
;P 0929  1       Load_Select_Com,  Load_Select_Com, Do_Nothing,
;P 0930  1       Load_Start_Com,  Load_Start Com,  Do_Nothing,
;P 0931  1       Diagnostic_Running, Diagnostic_Running, Do_Nothing,
;P 0932  1       Load_Error,  Load_Error,  Do_Nothing,
;P 0933  1       Start_Error,  Start_Error,  Do_Nothing,
;P 0934  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,
;P 0935  1       Preparation_Error, Preparation_Error, Do Nothing,  . [02]
;P 0936  1       Internal_Error,  Internal_Error,  Internal_Error
;   0937  1          ),
;   0938  1       ! This row is for the Command_Out typecode (Row 22)
;   0939  1
;P 0940  1     HS$Insert_N (
;P 0941  1       Diagnostic_Running,  Diagnostic_Running, Do_Nothing,
;P 0942  1       Internal_Error,  Internal_Error,  Internal_Error
;   0943  1          ),
```

```
;   0944  1       ! This row is for the Program_Info typecode (Row 23)
;   0945  1
; P 0946  1     HS$Insert_N (
; P 0947  1       Diagnostic_Running,  Start_Error,  Start_Error,
; P 0948  1       Start_Error,  Start_Error,  Do_Nothing,
; P 0949  1       Internal_Error,  Internal_Error,  Internal_Error
;   0950  1           ),
;   0951  1       ! This row is for the Start_Err typecode (Row 24)
;   0952  1
; P 0953  1     HS$Insert_N (
; P 0954  1       Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
; P 0955  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P 0956  1       Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 0957  1       Internal_Error,  Internal_Error,  Internal_Error
;   0958  1           ),
;   0959  1       ! This row is for the Sequence_Error typecode (Row 25)
;   0960  1
;   0961  1     HS$Fill_Full_Row,
;   0962  1       ! This row is for the CRD_AutoTest typecode (Row 26)
;   0963  1
; P 0964  1     HS$Insert_N (
; P 0965  1       Diagnostic_Running,  Preparation_Error, Preparation_Error,
; P 0966  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,  ! [02]
; P 0967  1       Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 0968  1       Internal_Error,  Internal_Error,  Internal_Error
;   0969  1           ),
;   0970  1       ! This row is for the ErrPrep typecode (Row 27)
;   0971  1
; P 0972  1     HS$Insert_N (
; P 0973  1       Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
; P 0974  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P 0975  1       Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 0976  1       Internal_Error,  Internal_Error,  Internal_Error
;   0977  1           ),
;   0978  1       ! This row is for the Param_Error typecode (Row 28)
;   0979  1
; P 0980  1     HS$Insert_N (
; P 0981  1       Internal_Error,  Internal_Error,  Internal_Error
;   0982  1           ),
;   0983  1       ! This row is for the DS_Start typecode (Row 29)
;   0984  1
; P 0985  1     HS$Insert_N (
; P 0986  1       Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
; P 0987  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P 0988  1       Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 0989  1       Internal_Error,  Internal_Error,  Internal_Error
;   0990  1           ),
;   0991  1       ! This row is for the Script_Pnf typecode (Row 30)
;   0992  1
; P 0993  1     HS$Insert_N (
; P 0994  1       Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
; P 0995  1       Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P 0996  1       Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 0997  1       Internal_Error,  Internal_Error,  Internal_Error
;   0998  1           ),
```

```
;   0999  1      ! This row is for the Script_Skip typecode (Row 31)
;   1000  1
; P 1001  1    HS$Insert_N (
; P 1002  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
; P 1003  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P 1004  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 1005  1      Internal_Error,  Internal_Error,  Internal_Error
;   1006  1         ),
;   1007  1      ! This row is for the Script_Prompt typecode (Row 32)
;   1008  1
; P 1009  1    HS$Insert_N (
; P 1010  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
; P 1011  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P 1012  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 1013  1      Internal_Error,  Internal_Error,  Internal_Error
;   1014  1         ),
;   1015  1      ! This row is for the Script_Echo typecode (Row 33)
;   1016  1
; P 1017  1    HS$Insert_N (
; P 1018  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
; P 1019  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P 1020  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 1021  1      Internal_Error,  Internal_Error,  Internal_Error
;   1022  1         ),
;   1023  1      ! This row is for the QIO_NoDriver typecode (Row 34)
;   1024  1
; P 1025  1    HS$Insert_N (
; P 1026  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
; P 1027  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P 1028  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 1029  1      Internal_Error,  Internal_Error,  Internal_Error
;   1030  1         ),
;   1031  1      ! This row is for the QIO_WrongVer typecode (Row 35)
;   1032  1
; P 1033  1    HS$Insert_N (
; P 1034  1      Diagnostic_Running,  Diagnostic_Error, Diagnostic_Error,
; P 1035  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,
; P 1036  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 1037  1      Internal_Error,  Internal_Error,  Internal_Error
;   1038  1         ),
;   1039  1      ! This row is for the QIO_InvAdp typecode (Row 36)
;   1040  1
; P 1041  1    HS$Insert_N (
; P 1042  1      Start_Error,  Start_Error,  Do_Nothing,
; P 1043  1      Diagnostic_Error, Diagnostic_Error, Do_Nothing,  ! [02]
; P 1044  1      Preparation_Error, Preparation_Error, Do_Nothing,  ! [02]
; P 1045  1      Internal_Error,  Internal_Error,  Internal_Error
;   1046  1         )
;   1047  1      ! This row is for the Start_List typecode (Row 37)
;   1048  1
;   1049  1    ) : HS_State_Table_Structure [CRD$K_Numb_TypeCodes, HS$K_Numb_States];
```

```
;   1050  1 END     ! End of MENSTATE module
;   1051  0 ELUDOM


            .TITLE  MENSTATE *** MENSTATE Module
            .IDENT  \V01.4\

            .PSECT  WORK,NOEXE,2

            00000 CRD$GL_PREVIOUS_STATE::
            .BLKB   4
            00004 CRD$GL_MM_CURRENT_STATE::
            .BLKB   4
            00008 CRD$GL_TQ_CURRENT_STATE::
            .BLKB   4
            0000C CRD$GL_HS_CURRENT_STATE::
            .BLKB   4
            00010 CRD$GL_TQ_CURRENT_TQ_ENTRY::
            .BLKB   4
            00014 CRD$GA_HS_CURRENT_DDB_PTR::
            .BLKB   4
            00018 CRD$GA_HS_CURRENT_HSB_PTR::
            .BLKB   4
            0001C CRD$GL_HS_CURRENT_HSB_NUMB::
            .BLKB   4
            00020 CRD$GB_CURRENT_STATE_TABLE::
            .BLKB   1

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

    45  54  41  54  53  4E  45  4D  08  00000 P.AAA:  .ASCII  <8>\MENSTATE\          ;
            00009   .BLKB   3
        0012  0012  0000C P.AAB:  .WORD   18, 18          ;
        0012  0012  00010     .WORD   18, 18          ;
        0012  0012  00014     .WORD   18, 18          ;
        0012  0012  00018     .WORD   18, 18          ;
        0012  0012  0001C     .WORD   18, 18          ;
        0012  0012  00020     .WORD   18, 18          ;
        0013  0013  00024     .WORD   19, 19          ;
        0013  0013  00028     .WORD   19, 19          ;
        0013  0013  0002C     .WORD   19, 19          ;
        0009  0000  00030     .WORD   0, 9            ;
        0012  0012  00034     .WORD   18, 18          ;
        0012  0012  00038     .WORD   18, 18          ;
        0012  0012  0003C     .WORD   18, 18          ;
        0012  0012  00040     .WORD   18, 18          ;
        0012  0012  00044     .WORD   18, 18          ;
        0012  0012  00048     .WORD   18, 18          ;
        0012  0012  0004C     .WORD   18, 18          ;
        0012  0012  00050     .WORD   18, 18          ;
        0012  0012  00054     .WORD   18, 18          ;
        0012  0012  00058     .WORD   18, 18          ;
        0012  0012  0005C     .WORD   18, 18          ;
        0012  0012  00060     .WORD   18, 18          ;
        0012  0012  00064     .WORD   18, 18          ;
```

```
0005  0001  00068     .WORD   1, 5            ;
0012  0012  0006C     .WORD   18, 18          ;
0006  0005  00070     .WORD   5, 6            ;
0007  0006  00074     .WORD   6, 7            ;
0008  0007  00078     .WORD   7, 8            ;
0009  0008  0007C     .WORD   8, 9            ;
000A  0001  00080     .WORD   1, 10           ;
000B  000A  00084     .WORD   10, 11          ;
000D  000B  00088     .WORD   11, 13          ;
0012  0012  0008C     .WORD   18, 18          ;
000F  000D  00090     .WORD   13, 15          ;
0012  0012  00094     .WORD   18, 18          ;
0010  000F  00098     .WORD   15, 16          ;
000F  0010  0009C     .WORD   16, 15          ;
0012  0012  000A0     .WORD   18, 18          ;
0012  0012  000A4     .WORD   18, 18          ;
0012  0012  000A8     .WORD   18, 18          ;
0012  0012  000AC     .WORD   18, 18          ;
0012  0012  000B0     .WORD   18, 18          ;
0012  0012  000B4     .WORD   18, 18          ;
0012  0012  000B8     .WORD   18, 18          ;
0012  0012  000BC     .WORD   18, 18          ;
0012  0012  000C0     .WORD   18, 18          ;
0012  0012  000C4     .WORD   18, 18          ;
0012  0012  000C8     .WORD   18, 18          ;
0012  0012  000CC     .WORD   18, 18          ;
0013  0013  000D0     .WORD   19, 19          ;
0012  0012  000D4     .WORD   18, 18          ;
0012  0012  000D8     .WORD   18, 18          ;
0012  0012  000DC     .WORD   18, 18          ;
0012  0012  000E0     .WORD   18, 18          ;
0012  0012  000E4     .WORD   18, 18          ;
0012  0012  000E8     .WORD   18, 18          ;
0010  0011  000EC     .WORD   17, 16          ;
0012  0012  000F0     .WORD   18, 18          ;
0012  0012  000F4     .WORD   18, 18          ;
0012  0012  000F8     .WORD   18, 18          ;
0012  0012  000FC     .WORD   18, 18          ;
0012  0012  00100     .WORD   18, 18          ;
0012  0012  00104     .WORD   18, 18          ;
0012  0012  00108     .WORD   18, 18          ;
0012  0012  0010C     .WORD   18, 18          ;
0012  0012  00110     .WORD   18, 18          ;
0013  0013  00114     .WORD   19, 19          ;
0013  0013  00118     .WORD   19, 19          ;
0013  0013  0011C     .WORD   19, 19          ;
0013  0013  00120     .WORD   19, 19          ;
0012  0012  00124     .WORD   18, 18          ;
0012  0012  00128     .WORD   18, 18          ;
0012  0012  0012C     .WORD   18, 18          ;
0012  0012  00130     .WORD   18, 18          ;
0012  0012  00134     .WORD   18, 18          ;
0012  0012  00138     .WORD   18, 18          ;
0012  0012  0013C     .WORD   18, 18          ;
0012  0012  00140     .WORD   18, 18          ;
```

```
0012  0012  00144     .WORD   18, 18          ;
0012  0012  00148     .WORD   18, 18          ;
0012  0012  0014C     .WORD   18, 18          ;
0012  0012  00150     .WORD   18, 18          ;
0012  0012  00154     .WORD   18, 18          ;
0012  0012  00158     .WORD   18, 18          ;
0012  0012  0015C     .WORD   18, 18          ;
0012  0012  00160     .WORD   18, 18          ;
0012  0012  00164     .WORD   18, 18          ;
0012  0012  00168     .WORD   18, 18          ;
0012  0012  0016C     .WORD   18, 18          ;
0013  0013  00170     .WORD   19, 19          ;
0012  0012  00174     .WORD   18, 18          ;
0012  0012  00178     .WORD   18, 18          ;
0012  0012  0017C     .WORD   18, 18          ;
0012  0012  00180     .WORD   18, 18          ;
0012  0012  00184     .WORD   18, 18          ;
0012  0012  00188     .WORD   18, 18          ;
0012  0012  0018C     .WORD   18, 18          ;
0012  0012  00190     .WORD   18, 18          ;
0012  0012  00194     .WORD   18, 18          ;
0012  0012  00198     .WORD   18, 18          ;
0012  0012  0019C     .WORD   18, 18          ;
0012  0012  001A0     .WORD   18, 18          ;
0012  0012  001A4     .WORD   18, 18          ;
0012  0012  001A8     .WORD   18, 18          ;
0012  0012  001AC     .WORD   18, 18          ;
0012  0012  001B0     .WORD   18, 18          ;
0012  0012  001B4     .WORD   18, 18          ;
0012  0012  001B8     .WORD   18, 18          ;
0012  0012  001BC     .WORD   18, 18          ;
0013  0013  001C0     .WORD   19, 19          ;
0012  0012  001C4     .WORD   18, 18          ;
0012  0012  001C8     .WORD   18, 18          ;
0012  0012  001CC     .WORD   18, 18          ;
0012  0012  001D0     .WORD   18, 18          ;
0012  0012  001D4     .WORD   18, 18          ;
0012  0012  001D8     .WORD   18, 18          ;
0012  0012  001DC     .WORD   18, 18          ;
0012  0012  001E0     .WORD   18, 18          ;
0012  0012  001E4     .WORD   18, 18          ;
0012  0012  001E8     .WORD   18, 18          ;
0012  0012  001EC     .WORD   18, 18          ;
0012  0012  001F0     .WORD   18, 18          ;
0012  0012  001F4     .WORD   18, 18          ;
0012  0012  001F8     .WORD   18, 18          ;
0012  0012  001FC     .WORD   18, 18          ;
0012  0012  00200     .WORD   18, 18          ;
0012  0012  00204     .WORD   18, 18          ;
0012  0012  00208     .WORD   18, 18          ;
0012  0012  0020C     .WORD   18, 18          ;
0013  0013  00210     .WORD   19, 19          ;
0012  0012  00214     .WORD   18, 18          ;
0012  0012  00218     .WORD   18, 18          ;
0012  0012  0021C     .WORD   18, 18          ;
```

```
0012  0012  00220     .WORD   18, 18          ;
0012  0012  00224     .WORD   18, 18          ;
0012  0012  00228     .WORD   18, 18          ;
0012  0012  0022C     .WORD   18, 18          ;
0012  0012  00230     .WORD   18, 18          ;
0012  0012  00234     .WORD   18, 18          ;
0012  0012  00238     .WORD   18, 18          ;
0012  0012  0023C     .WORD   18, 18          ;
0012  0012  00240     .WORD   18, 18          ;
0012  0012  00244     .WORD   18, 18          ;
0012  0012  00248     .WORD   18, 18          ;
0012  0012  0024C     .WORD   18, 18          ;
0012  0012  00250     .WORD   18, 18          ;
0012  0012  00254     .WORD   18, 18          ;
0012  0012  00258     .WORD   18, 18          ;
0012  0012  0025C     .WORD   18, 18          ;
0013  0013  00260     .WORD   19, 19          ;
0012  0012  00264     .WORD   18, 18          ;
0012  0012  00268     .WORD   18, 18          ;
0012  0012  0026C     .WORD   18, 18          ;
0012  0012  00270     .WORD   18, 18          ;
0012  0012  00274     .WORD   18, 18          ;
0012  0012  00278     .WORD   18, 18          ;
0012  0012  0027C     .WORD   18, 18          ;
0012  0012  00280     .WORD   18, 18          ;
0012  0012  00284     .WORD   18, 18          ;
0012  0012  00288     .WORD   18, 18          ;
0012  0012  0028C     .WORD   18, 18          ;
0012  0012  00290     .WORD   18, 18          ;
0012  0012  00294     .WORD   18, 18          ;
0012  0012  00298     .WORD   18, 18          ;
0012  0012  0029C     .WORD   18, 18          ;
0012  0012  002A0     .WORD   18, 18          ;
0012  0012  002A4     .WORD   18, 18          ;
0012  0012  002A8     .WORD   18, 18          ;
0012  0012  002AC     .WORD   18, 18          ;
0013  0013  002B0     .WORD   19, 19          ;
0012  0012  002B4     .WORD   18, 18          ;
0012  0012  002B8     .WORD   18, 18          ;
0012  0012  002BC     .WORD   18, 18          ;
0012  0012  002C0     .WORD   18, 18          ;
0012  0012  002C4     .WORD   18, 18          ;
0012  0012  002C8     .WORD   18, 18          ;
0012  0012  002CC     .WORD   18, 18          ;
0012  0012  002D0     .WORD   18, 18          ;
0012  0012  002D4     .WORD   18, 18          ;
0012  0012  002D8     .WORD   18, 18          ;
0012  0012  002DC     .WORD   18, 18          ;
0012  0012  002E0     .WORD   18, 18          ;
0012  0012  002E4     .WORD   18, 18          ;
0012  0012  002E8     .WORD   18, 18          ;
0012  0012  002EC     .WORD   18, 18          ;
0012  0012  002F0     .WORD   18, 18          ;
0012  0012  002F4     .WORD   18, 18          ;
0012  0012  002F8     .WORD   18, 18          ;
```

```
      0012  0012  002FC     .WORD   18, 18           ;
      0013  0013  00300     .WORD   19, 19           ;
      0012  0012  00304     .WORD   18, 18           ;
      0012  0012  00308     .WORD   18, 18           ;
      0012  0012  0030C     .WORD   18, 18           ;
      0012  0012  00310     .WORD   18, 18           ;
      0012  0012  00314     .WORD   18, 18           ;
      0012  0012  00318     .WORD   18, 18           ;
      0012  0012  0031C     .WORD   18, 18           ;
      0012  0012  00320     .WORD   18, 18           ;
      0012  0012  00324     .WORD   18, 18           ;
      0012  0012  00328     .WORD   18, 18           ;
      0012  0012  0032C     .WORD   18, 18           ;
      0012  0012  00330     .WORD   18, 18           ;
      0012  0012  00334     .WORD   18, 18           ;
      0012  0012  00338     .WORD   18, 18           ;
      0012  0012  0033C     .WORD   18, 18           ;
      0012  0012  00340     .WORD   18, 18           ;
      0012  0012  00344     .WORD   18, 18           ;
      0012  0012  00348     .WORD   18, 18           ;
      0012  0012  0034C     .WORD   18, 18           ;
      0013  0013  00350     .WORD   19, 19           ;
      0012  0012  00354     .WORD   18, 18           ;
      0012  0012  00358     .WORD   18, 18           ;
      0012  0012  0035C     .WORD   18, 18           ;
      0012  0012  00360     .WORD   18, 18           ;
      0012  0012  00364     .WORD   18, 18           ;
      0012  0012  00368     .WORD   18, 18           ;
      0012  0012  0036C     .WORD   18, 18           ;
      0012  0012  00370     .WORD   18, 18           ;
      0012  0012  00374     .WORD   18, 18           ;
      0012  0012  00378     .WORD   18, 18           ;
      0012  0012  0037C     .WORD   18, 18           ;
      0012  0012  00380     .WORD   18, 18           ;
      0012  0012  00384     .WORD   18, 18           ;
      0012  0012  00388     .WORD   18, 18           ;
      0012  0012  0038C     .WORD   18, 18           ;
      0012  0012  00390     .WORD   18, 18           ;
      0012  0012  00394     .WORD   18, 18           ;
      0012  0012  00398     .WORD   18, 18           ;
      0012  0012  0039C     .WORD   18, 18           ;
      0013  0013  003A0     .WORD   19, 19           ;
      0012  0012  003A4     .WORD   18, 18           ;
      0012  0012  003A8     .WORD   18, 18           ;
      0012  0012  003AC     .WORD   18, 18           ;
      0012  0012  003B0     .WORD   18, 18           ;
      0012  0012  003B4     .WORD   18, 18           ;
      0012  0012  003B8     .WORD   18, 18           ;
      0012  0012  003BC     .WORD   18, 18           ;
      0012  0012  003C0     .WORD   18, 18           ;
      0012  0012  003C4     .WORD   18, 18           ;
      0012  0012  003C8     .WORD   18, 18           ;
      0012  0012  003CC     .WORD   18, 18           ;
      0012  0012  003D0     .WORD   18, 18           ;
      0012  0012  003D4     .WORD   18, 18           ;
```

```
0012  0012  003D8     .WORD   18, 18          ;
0012  0012  003DC     .WORD   18, 18          ;
0012  0012  003E0     .WORD   18, 18          ;
0012  0012  003E4     .WORD   18, 18          ;
0012  0012  003E8     .WORD   18, 18          ;
0012  0012  003EC     .WORD   18, 18          ;
0013  0013  003F0     .WORD   19, 19          ;
0012  0012  003F4     .WORD   18, 18          ;
0012  0012  003F8     .WORD   18, 18          ;
0012  0012  003FC     .WORD   18, 18          ;
0012  0012  00400     .WORD   18, 18          ;
0012  0012  00404     .WORD   18, 18          ;
0012  0012  00408     .WORD   18, 18          ;
0012  0012  0040C     .WORD   18, 18          ;
0012  0012  00410     .WORD   18, 18          ;
0012  0012  00414     .WORD   18, 18          ;
0012  0012  00418     .WORD   18, 18          ;
0012  0012  0041C     .WORD   18, 18          ;
0012  0012  00420     .WORD   18, 18          ;
0012  0012  00424     .WORD   18, 18          ;
0012  0012  00428     .WORD   18, 18          ;
0012  0012  0042C     .WORD   18, 18          ;
0012  0012  00430     .WORD   18, 18          ;
0012  0012  00434     .WORD   18, 18          ;
0012  0012  00438     .WORD   18, 18          ;
0012  0012  0043C     .WORD   18, 18          ;
0013  0013  00440     .WORD   19, 19          ;
0012  0012  00444     .WORD   18, 18          ;
0012  0012  00448     .WORD   18, 18          ;
0012  0012  0044C     .WORD   18, 18          ;
0012  0012  00450     .WORD   18, 18          ;
0012  0012  00454     .WORD   18, 18          ;
0012  0012  00458     .WORD   18, 18          ;
0012  0012  0045C     .WORD   18, 18          ;
0012  0012  00460     .WORD   18, 18          ;
0012  0012  00464     .WORD   18, 18          ;
0012  0012  00468     .WORD   18, 18          ;
0012  0012  0046C     .WORD   18, 18          ;
0012  0012  00470     .WORD   18, 18          ;
0012  0012  00474     .WORD   18, 18          ;
0012  0012  00478     .WORD   18, 18          ;
0012  0012  0047C     .WORD   18, 18          ;
0012  0012  00480     .WORD   18, 18          ;
0012  0012  00484     .WORD   18, 18          ;
0012  0012  00488     .WORD   18, 18          ;
0012  0012  0048C     .WORD   18, 18          ;
0009  0000  00490     .WORD   0, 9          ;
0012  0012  00494     .WORD   18, 18          ;
0012  0012  00498     .WORD   18, 18          ;
0012  0012  0049C     .WORD   18, 18          ;
0012  0012  004A0     .WORD   18, 18          ;
0012  0012  004A4     .WORD   18, 18          ;
0012  0012  004A8     .WORD   18, 18          ;
0012  0012  004AC     .WORD   18, 18          ;
0012  0012  004B0     .WORD   18, 18          ;
```

```
0012  0012  004B4     .WORD   18, 18          ;
0012  0012  004B8     .WORD   18, 18          ;
0012  0012  004BC     .WORD   18, 18          ;
0012  0012  004C0     .WORD   18, 18          ;
0012  0012  004C4     .WORD   18, 18          ;
0012  0012  004C8     .WORD   18, 18          ;
0012  0012  004CC     .WORD   18, 18          ;
0012  0012  004D0     .WORD   18, 18          ;
0012  0012  004D4     .WORD   18, 18          ;
0012  0012  004D8     .WORD   18, 18          ;
0012  0012  004DC     .WORD   18, 18          ;
0009  0000  004E0     .WORD   0, 9           ;
0012  0012  004E4     .WORD   18, 18          ;
0012  0012  004E8     .WORD   18, 18          ;
0012  0012  004EC     .WORD   18, 18          ;
0012  0012  004F0     .WORD   18, 18          ;
0012  0012  004F4     .WORD   18, 18          ;
0012  0012  004F8     .WORD   18, 18          ;
0012  0012  004FC     .WORD   18, 18          ;
0012  0012  00500     .WORD   18, 18          ;
0012  0012  00504     .WORD   18, 18          ;
0012  0012  00508     .WORD   18, 18          ;
0012  0012  0050C     .WORD   18, 18          ;
0012  0012  00510     .WORD   18, 18          ;
0012  0012  00514     .WORD   18, 18          ;
0012  0012  00518     .WORD   18, 18          ;
0012  0012  0051C     .WORD   18, 18          ;
0012  0012  00520     .WORD   18, 18          ;
0012  0012  00524     .WORD   18, 18          ;
0012  0012  00528     .WORD   18, 18          ;
0012  0012  0052C     .WORD   18, 18          ;
0009  0000  00530     .WORD   0, 9           ;
0012  0012  00534     .WORD   18, 18          ;
0012  0012  00538     .WORD   18, 18          ;
0012  0012  0053C     .WORD   18, 18          ;
0012  0012  00540     .WORD   18, 18          ;
0012  0012  00544     .WORD   18, 18          ;
0012  0012  00548     .WORD   18, 18          ;
0012  0012  0054C     .WORD   18, 18          ;
0012  0012  00550     .WORD   18, 18          ;
0012  0012  00554     .WORD   18, 18          ;
0012  0012  00558     .WORD   18, 18          ;
0012  0012  0055C     .WORD   18, 18          ;
0012  0012  00560     .WORD   18, 18          ;
0012  0012  00564     .WORD   18, 18          ;
0012  0012  00568     .WORD   18, 18          ;
0012  0012  0056C     .WORD   18, 18          ;
0012  0012  00570     .WORD   18, 18          ;
0012  0012  00574     .WORD   18, 18          ;
0012  0012  00578     .WORD   18, 18          ;
0012  0012  0057C     .WORD   18, 18          ;
0009  0000  00580     .WORD   0, 9           ;
0012  0012  00584     .WORD   18, 18          ;
0012  0012  00588     .WORD   18, 18          ;
0012  0012  0058C     .WORD   18, 18          ;
```

```
0012  0012  00590     .WORD   18, 18          ;
0012  0012  00594     .WORD   18, 18          ;
0012  0012  00598     .WORD   18, 18          ;
0012  0012  0059C     .WORD   18, 18          ;
0012  0012  005A0     .WORD   18, 18          ;
0012  0012  005A4     .WORD   18, 18          ;
0012  0012  005A8     .WORD   18, 18          ;
0012  0012  005AC     .WORD   18, 18          ;
0012  0012  005B0     .WORD   18, 18          ;
0012  0012  005B4     .WORD   18, 18          ;
0012  0012  005B8     .WORD   18, 18          ;
0012  0012  005BC     .WORD   18, 18          ;
0012  0012  005C0     .WORD   18, 18          ;
0012  0012  005C4     .WORD   18, 18          ;
0012  0012  005C8     .WORD   18, 18          ;
0012  0012  005CC     .WORD   18, 18          ;
0013  0013  005D0     .WORD   19, 19          ;
0012  0012  005D4     .WORD   18, 18          ;
0012  0012  005D8     .WORD   18, 18          ;
0012  0012  005DC     .WORD   18, 18          ;
0012  0012  005E0     .WORD   18, 18          ;
0012  0012  005E4     .WORD   18, 18          ;
0012  0012  005E8     .WORD   18, 18          ;
0012  0012  005EC     .WORD   18, 18          ;
0012  0012  005F0     .WORD   18, 18          ;
0012  0012  005F4     .WORD   18, 18          ;
0012  0012  005F8     .WORD   18, 18          ;
0012  0012  005FC     .WORD   18, 18          ;
0012  0012  00600     .WORD   18, 18          ;
0012  0012  00604     .WORD   18, 18          ;
0012  0012  00608     .WORD   18, 18          ;
0012  0012  0060C     .WORD   18, 18          ;
0012  0012  00610     .WORD   18, 18          ;
0012  0012  00614     .WORD   18, 18          ;
0012  0012  00618     .WORD   18, 18          ;
0012  0012  0061C     .WORD   18, 18          ;
0013  0013  00620     .WORD   19, 19          ;
0012  0012  00624     .WORD   18, 18          ;
0012  0012  00628     .WORD   18, 18          ;
0012  0012  0062C     .WORD   18, 18          ;
0012  0012  00630     .WORD   18, 18          ;
0012  0012  00634     .WORD   18, 18          ;
0012  0012  00638     .WORD   18, 18          ;
0012  0012  0063C     .WORD   18, 18          ;
0012  0012  00640     .WORD   18, 18          ;
0012  0012  00644     .WORD   18, 18          ;
0012  0012  00648     .WORD   18, 18          ;
0012  0012  0064C     .WORD   18, 18          ;
0012  0012  00650     .WORD   18, 18          ;
0012  0012  00654     .WORD   18, 18          ;
0012  0012  00658     .WORD   18, 18          ;
0012  0012  0065C     .WORD   18, 18          ;
0012  0012  00660     .WORD   18, 18          ;
0012  0012  00664     .WORD   18, 18          ;
0012  0012  00668     .WORD   18, 18          ;
```

```
0012  0012  0066C     .WORD   18, 18          ;
0013  0013  00670     .WORD   19, 19          ;
0012  0012  00674     .WORD   18, 18          ;
0012  0012  00678     .WORD   18, 18          ;
0012  0012  0067C     .WORD   18, 18          ;
0012  0012  00680     .WORD   18, 18          ;
0012  0012  00684     .WORD   18, 18          ;
0012  0012  00688     .WORD   18, 18          ;
0012  0012  0068C     .WORD   18, 18          ;
0012  0012  00690     .WORD   18, 18          ;
0012  0012  00694     .WORD   18, 18          ;
0012  0012  00698     .WORD   18, 18          ;
0012  0012  0069C     .WORD   18, 18          ;
0012  0012  006A0     .WORD   18, 18          ;
0012  0012  006A4     .WORD   18, 18          ;
0012  0012  006A8     .WORD   18, 18          ;
0012  0012  006AC     .WORD   18, 18          ;
0012  0012  006B0     .WORD   18, 18          ;
0013  0013  006B4     .WORD   19, 19          ;
0013  0013  006B8     .WORD   19, 19          ;
0013  0013  006BC     .WORD   19, 19          ;
0013  0013  006C0     .WORD   19, 19          ;
0012  0012  006C4     .WORD   18, 18          ;
0012  0012  006C8     .WORD   18, 18          ;
0012  0012  006CC     .WORD   18, 18          ;
0012  0012  006D0     .WORD   18, 18          ;
0012  0012  006D4     .WORD   18, 18          ;
0012  0012  006D8     .WORD   18, 18          ;
0012  0012  006DC     .WORD   18, 18          ;
0012  0012  006E0     .WORD   18, 18          ;
0012  0012  006E4     .WORD   18, 18          ;
0012  0012  006E8     .WORD   18, 18          ;
0012  0012  006EC     .WORD   18, 18          ;
0012  0012  006F0     .WORD   18, 18          ;
0012  0012  006F4     .WORD   18, 18          ;
0012  0012  006F8     .WORD   18, 18          ;
0012  0012  006FC     .WORD   18, 18          ;
0012  0012  00700     .WORD   18, 18          ;
0013  0013  00704     .WORD   19, 19          ;
0013  0013  00708     .WORD   19, 19          ;
0013  0013  0070C     .WORD   19, 19          ;
0013  0013  00710     .WORD   19, 19          ;
0012  0012  00714     .WORD   18, 18          ;
0012  0012  00718     .WORD   18, 18          ;
0012  0012  0071C     .WORD   18, 18          ;
0012  0012  00720     .WORD   18, 18          ;
0012  0012  00724     .WORD   18, 18          ;
0012  0012  00728     .WORD   18, 18          ;
0012  0012  0072C     .WORD   18, 18          ;
0012  0012  00730     .WORD   18, 18          ;
0012  0012  00734     .WORD   18, 18          ;
0012  0012  00738     .WORD   18, 18          ;
0012  0012  0073C     .WORD   18, 18          ;
0012  0012  00740     .WORD   18, 18          ;
0012  0012  00744     .WORD   18, 18          ;
```

```
      0012  0012  00748     .WORD   18, 18          ;
      0012  0012  0074C     .WORD   18, 18          ;
      0012  0012  00750     .WORD   18, 18          ;
      0012  0012  00754     .WORD   18, 18          ;
      0012  0012  00758     .WORD   18, 18          ;
      0012  0012  0075C     .WORD   18, 18          ;
      0009  0000  00760     .WORD   0, 9           ;
      0012  0012  00764     .WORD   18, 18          ;
      0012  0012  00768     .WORD   18, 18          ;
      0012  0012  0076C     .WORD   18, 18          ;
      0012  0012  00770     .WORD   18, 18          ;
      0012  0012  00774     .WORD   18, 18          ;
      0012  0012  00778     .WORD   18, 18          ;
      0012  0012  0077C     .WORD   18, 18          ;
      0012  0012  00780     .WORD   18, 18          ;
      0012  0012  00784     .WORD   18, 18          ;
      0012  0012  00788     .WORD   18, 18          ;
      0012  0012  0078C     .WORD   18, 18          ;
      0012  0012  00790     .WORD   18, 18          ;
      0012  0012  00794     .WORD   18, 18          ;
      0012  0012  00798     .WORD   18, 18          ;
      0012  0012  0079C     .WORD   18, 18          ;
      0012  0012  007A0     .WORD   18, 18          ;
      0012  0012  007A4     .WORD   18, 18          ;
      0012  0012  007A8     .WORD   18, 18          ;
      0012  0012  007AC     .WORD   18, 18          ;
      0013  0013  007B0     .WORD   19, 19          ;
      0012  0012  007B4     .WORD   18, 18          ;
      0012  0012  007B8     .WORD   18, 18          ;
      0012  0012  007BC     .WORD   18, 18          ;
      0012  0012  007C0     .WORD   18, 18          ;
      0012  0012  007C4     .WORD   18, 18          ;
      0012  0012  007C8     .WORD   18, 18          ;
      0012  0012  007CC     .WORD   18, 18          ;
      0012  0012  007D0     .WORD   18, 18          ;
      0012  0012  007D4     .WORD   18, 18          ;
      0012  0012  007D8     .WORD   18, 18          ;
      0012  0012  007DC     .WORD   18, 18          ;
      0012  0012  007E0     .WORD   18, 18          ;
      0012  0012  007E4     .WORD   18, 18          ;
      0012  0012  007E8     .WORD   18, 18          ;
      0012  0012  007EC     .WORD   18, 18          ;
      0012  0012  007F0     .WORD   18, 18          ;
      0012  0012  007F4     .WORD   18, 18          ;
      0012  0012  007F8     .WORD   18, 18          ;
      0012  0012  007FC     .WORD   18, 18          ;
      0013  0013  00800     .WORD   19, 19          ;
      0012  0012  00804     .WORD   18, 18          ;
      0012  0012  00808     .WORD   18, 18          ;
      0012  0012  0080C     .WORD   18, 18          ;
      0012  0012  00810     .WORD   18, 18          ;
      0012  0012  00814     .WORD   18, 16          ;
      0012  0012  00818     .WORD   18, 18          ;
      0012  0012  0081C     .WORD   18, 18          ;
      0012  0012  00820     .WORD   18, 18          ;
```

```
0012  0012  00824     .WORD   18, 18           ;
0012  0012  00828     .WORD   18, 18           ;
0012  0012  0082C     .WORD   18, 18           ;
0012  0012  00830     .WORD   18, 18           ;
0012  0012  00834     .WORD   18, 18           ;
0012  0012  00838     .WORD   18, 18           ;
0012  0012  0083C     .WORD   18, 18           ;
0012  0012  00840     .WORD   18, 18           ;
0012  0012  00844     .WORD   18, 18           ;
0012  0012  00848     .WORD   18, 18           ;
0012  0012  0084C     .WORD   18, 18           ;
0012  0012  00850     .WORD   18, 18           ;
0012  0012  00854     .WORD   18, 18           ;
0012  0012  00858     .WORD   18, 18           ;
0012  0012  0085C     .WORD   18, 18           ;
0012  0012  00860     .WORD   18, 18           ;
0012  0012  00864     .WORD   18, 18           ;
0012  0012  00868     .WORD   18, 18           ;
0012  0012  0086C     .WORD   18, 18           ;
0012  0012  00870     .WORD   18, 18           ;
0012  0012  00874     .WORD   18, 18           ;
0012  0012  00878     .WORD   18, 18           ;
0012  0012  0087C     .WORD   18, 18           ;
0012  0012  00880     .WORD   18, 18           ;
0012  0012  00884     .WORD   18, 18           ;
0012  0012  00888     .WORD   18, 18           ;
0012  0012  0088C     .WORD   18, 18           ;
0012  0012  00890     .WORD   18, 18           ;
0012  0012  00894     .WORD   18, 18           ;
0012  0012  00898     .WORD   18, 18           ;
0012  0012  0089C     .WORD   18, 18           ;
0013  0013  008A0     .WORD   19, 19           ;
0012  0012  008A4     .WORD   18, 18           ;
0012  0012  008A8     .WORD   18, 18           ;
0012  0012  008AC     .WORD   18, 18           ;
0012  0012  008B0     .WORD   18, 18           ;
0012  0012  008B4     .WORD   18, 18           ;
0012  0012  008B8     .WORD   18, 18           ;
0012  0012  008BC     .WORD   18, 18           ;
0012  0012  008C0     .WORD   18, 18           ;
0012  0012  008C4     .WORD   18, 18           ;
0012  0012  008C8     .WORD   18, 18           ;
0012  0012  008CC     .WORD   18, 18           ;
0012  0012  008D0     .WORD   18, 18           ;
0012  0012  008D4     .WORD   18, 18           ;
0012  0012  008D8     .WORD   18, 18           ;
0012  0012  008DC     .WORD   18, 18           ;
0012  0012  008E0     .WORD   18, 18           ;
0012  0012  008E4     .WORD   18, 18           ;
0012  0012  008E8     .WORD   18, 18           ;
0012  0012  008EC     .WORD   18, 18           ;
0013  0013  008F0     .WORD   19, 19           ;
0012  0012  008F4     .WORD   18, 18           ;
0012  0012  008F8     .WORD   18, 18           ;
0012  0012  008FC     .WORD   18, 18           ;
```

```
0012  0012  00900     .WORD   18, 18          ;
0012  0012  00904     .WORD   18, 18          ;
0012  0012  00908     .WORD   18, 18          ;
0012  0012  0090C     .WORD   18, 18          ;
0012  0012  00910     .WORD   18, 18          ;
0012  0012  00914     .WORD   18, 18          ;
0012  0012  00918     .WORD   18, 18          ;
0012  0012  0091C     .WORD   18, 18          ;
0012  0012  00920     .WORD   18, 18          ;
0012  0012  00924     .WORD   18, 18          ;
0005  0000  00928     .WORD   0, 5           ;
0012  0012  0092C     .WORD   18, 18          ;
0012  0012  00930     .WORD   18, 18          ;
0012  0012  00934     .WORD   18, 18          ;
0012  0012  00938     .WORD   18, 18          ;
0012  0012  0093C     .WORD   18, 18          ;
0012  0012  00940     .WORD   18, 18          ;
0012  0012  00944     .WORD   18, 18          ;
0012  0012  00948     .WORD   18, 18          ;
0012  0012  0094C     .WORD   18, 18          ;
0012  0012  00950     .WORD   18, 18          ;
0012  0012  00954     .WORD   18, 18          ;
0012  0012  00958     .WORD   18, 18          ;
0012  0012  0095C     .WORD   18, 18          ;
0012  0012  00960     .WORD   18, 18          ;
0012  0012  00964     .WORD   18, 18          ;
0012  0012  00968     .WORD   18, 18          ;
0012  0012  0096C     .WORD   18, 18          ;
0012  0012  00970     .WORD   18, 18          ;
0012  0012  00974     .WORD   18, 18          ;
0012  0012  00978     .WORD   18, 18          ;
0012  0012  0097C     .WORD   18, 18          ;
0012  0012  00980     .WORD   18, 18          ;
0012  0012  00984     .WORD   18, 18          ;
0012  0012  00988     .WORD   18, 18          ;
0012  0012  0098C     .WORD   18, 18          ;
0012  0012  00990     .WORD   18, 18          ;
0012  0012  00994     .WORD   18, 18          ;
0012  0012  00998     .WORD   18, 18          ;
0012  0012  0099C     .WORD   18, 18          ;
0012  0012  009A0     .WORD   18, 18          ;
0012  0012  009A4     .WORD   18, 18          ;
0012  0012  009A8     .WORD   18, 18          ;
0012  0012  009AC     .WORD   18, 18          ;
0012  0012  009B0     .WORD   18, 18          ;
0012  0012  009B4     .WORD   18, 18          ;
0012  0012  009B8     .WORD   18, 18          ;
0012  0012  009BC     .WORD   18, 18          ;
0012  0012  009C0     .WORD   18, 18          ;
0012  0012  009C4     .WORD   18, 18          ;
0012  0012  009C8     .WORD   18, 18          ;
0012  0012  009CC     .WORD   18, 18          ;
0012  0012  009D0     .WORD   18, 18          ;
0012  0012  009D4     .WORD   18, 18          ;
0012  0012  009D8     .WORD   18, 18          ;
```

```
      0012  0012  009DC     .WORD   18, 18          ;
      0012  0012  009E0     .WORD   18, 18          ;
      0C12  0012  009E4     .WORD   18, 18          ;
      0012  0012  009E8     .WORD   18, 18          ;
      0012  0012  009EC     .WORD   18, 18          ;
      0012  0012  009F0     .WORD   18, 18          ;
      0012  0012  009F4     .WORD   18, 18          ;
      0012  0012  009F8     .WORD   18, 18          ;
      0012  0012  009FC     .WORD   18, 18          ;
      0012  0012  00A00     .WORD   18, 18          ;
      0012  0012  00A04     .WORD   18, 18          ;
      0012  0012  00A08     .WORD   18, 18          ;
      0012  0012  00A0C     .WORD   18, 18          ;
      0012  0012  00A10     .WORD   18, 18          ;
      0012  0012  00A14     .WORD   18, 18          ;
      0012  0012  00A18     .WORD   18, 18          ;
      0012  0012  00A1C     .WORD   18, 18          ;
      0012  0012  00A20     .WORD   18, 18          ;
      0012  0012  00A24     .WORD   18, 18          ;
      0012  0012  00A28     .WORD   18, 18          ;
      0012  0012  00A2C     .WORD   18, 18          ;
      0012  0012  00A30     .WORD   18, 18          ;
      0012  0012  00A34     .WORD   18, 18          ;
      0012  0012  00A38     .WORD   18, 18          ;
      0012  0012  00A3C     .WORD   18, 18          ;
      0012  0012  00A40     .WORD   18, 18          ;
      0012  0012  00A44     .WORD   18, 18          ;
      0012  0012  00A48     .WORD   18, 18          ;
      0010  0011  00A4C     .WORD   17, 16          ;
      0012  0012  00A50     .WORD   18, 18          ;
      0012  0012  00A54     .WORD   18, 18          ;
      0012  0012  00A58     .WORD   18, 18          ;
      0012  0012  00A5C     .WORD   18, 18          ;
      0012  0012  00A60     .WORD   18, 18          ;
      0012  0012  00A64     .WORD   18, 18          ;
      0012  0012  00A68     .WORD   18, 18          ;
      0012  0012  00A6C     .WORD   18, 18          ;
      0012  0012  00A70     .WORD   18, 18          ;
      0012  0012  00A74     .WORD   18, 18          ;
      0012  0012  00A78     .WORD   18, 18          ;
      0012  0012  00A7C     .WORD   18, 18          ;
      0012  0012  00A80     .WORD   18, 18          ;
      0012  0012  00A84     .WORD   18, 18          ;
      0012  0012  00A88     .WORD   18, 18          ;
      0012  0012  00A8C     .WORD   18, 18          ;
      0012  0012  00A90     .WORD   18, 18          ;
      0012  0012  00A94     .WORD   18, 18          ;
      0012  0012  00A98     .WORD   18, 18          ;
      0012  0012  00A9C     .WORD   18, 18          ;
      0012  0012  00AA0     .WORD   18, 18          ;
      0012  0012  00AA4     .WORD   18, 18          ;
      0012  0012  00AA8     .WORD   18, 18          ;
      0012  0012  00AAC     .WORD   18, 18          ;
      0012  0012  00AB0     .WORD   18, 18          ;
      0012  0012  00AB4     .WORD   18, 18          ;
```

```
0012  0012  00AB8     .WORD   18, 18          ;
0012  0012  00ABC     .WORD   18, 18          ;
0012  0012  00AC0     .WORD   18, 18          ;
0012  0012  00AC4     .WORD   18, 18          ;
0012  0012  00AC8     .WORD   18, 18          ;
0012  0012  00ACC     .WORD   18, 18          ;
0013  0013  00AD0     .WORD   19, 19          ;
0012  0012  00AD4     .WORD   18, 18          ;
0012  0012  00AD8     .WORD   18, 18          ;
0012  0012  00ADC     .WORD   18, 18          ;
0012  0012  00AE0     .WORD   18, 18          ;
0012  0012  00AE4     .WORD   18, 18          ;
0012  0012  00AE8     .WORD   18, 18          ;
0012  0012  00AEC     .WORD   18, 18          ;
0012  0012  00AF0     .WORD   18, 18          ;
0012  0012  00AF4     .WORD   18, 18          ;
0012  0012  00AF8     .WORD   18, 18          ;
0012  0012  00AFC     .WORD   18, 18          ;
0012  0012  00B00     .WORD   18, 18          ;
0012  0012  00B04     .WORD   18, 18          ;
0012  0012  00B08     .WORD   18, 18          ;
0012  0012  00B0C     .WORD   18, 18          ;
0012  0012  00B10     .WORD   18, 18          ;
0012  0012  00B14     .WORD   18, 18          ;
0012  0012  00B18     .WORD   18, 18          ;
0012  0012  00B1C     .WORD   18, 18          ;
0013  0013  00B20     .WORD   19, 19          ;
0012  0012  00B24     .WORD   18, 18          ;
0012  0012  00B28     .WORD   18, 18          ;
0012  0012  00B2C     .WORD   18, 18          ;
0012  0012  00B30     .WORD   18, 18          ;
0012  0012  00B34     .WORD   18, 18          ;
0012  0012  00B38     .WORD   18, 18          ;
0012  0012  00B3C     .WORD   18, 18          ;
0012  0012  00B40     .WORD   18, 18          ;
0012  0012  00B44     .WORD   18, 18          ;
0012  0012  00B48     .WORD   18, 18          ;
0012  0012  00B4C     .WORD   18, 18          ;
0012  0012  00B50     .WORD   18, 18          ;
0012  0012  00B54     .WORD   18, 18          ;
0012  0012  00B58     .WORD   18, 18          ;
0012  0012  00B5C     .WORD   18, 18          ;
0012  0012  00B60     .WORD   18, 18          ;
0012  0012  00B64     .WORD   18, 18          ;
0012  0012  00B68     .WORD   18, 18          ;
0012  0012  00B6C     .WORD   18, 18          ;
0013  0013  00B70     .WORD   19, 19          ;
0012  0012  00B74     .WORD   18, 18          ;
0012  0012  00B78     .WORD   18, 18          ;
0012  0012  00B7C     .WORD   18, 18          ;
0012  0012  00B80     .WORD   18, 18          ;
0012  0012  00B84     .WORD   18, 18          ;
0012  0012  00B88     .WORD   18, 18          ;
0012  0012  00B8C     .WORD   18, 18          ;
0012  0012  00B90     .WORD   18, 18          ;
```

```
0012  0012  00B94     .WORD   18, 18          ;
0012  0012  00B98     .WORD   18, 18          ;
0012  0012  00B9C     .WORD   18, 18          ;
0012  0012  00BA0     .WORD   18, 18          ;
0012  0012  00BA4     .WORD   18, 18          ;
0012  0012  00BA8     .WORD   18, 18          ;
0012  0012  00BAC     .WORD   18, 18          ;
0012  0012  00BB0     .WORD   18, 18          ;
0012  0012  00BB4     .WORD   18, 18          ;
0012  0012  00BB8     .WORD   18, 18          ;
0012  0012  00BBC     .WORD   18, 18          ;
0012  0012  00BC0     .WORD   18, 18          ;
0012  0012  00BC4     .WORD   18, 18          ;
0012  0012  00BC8     .WORD   18, 18          ;
0012  0012  00BCC     .WORD   18, 18          ;
0012  0012  00BD0     .WORD   18, 18          ;
0012  0012  00BD4     .WORD   18, 18          ;
0012  0012  00BD8     .WORD   18, 18          ;
0012  0012  00BDC     .WORD   18, 18          ;
0012  0012  00BE0     .WORD   18, 18          ;
0012  0012  00BE4     .WORD   18, 18          ;
0012  0012  00BE8     .WORD   18, 18          ;
0009  0009  00BEC P.AAC:  .WORD   9, 9            ;
0009  0009  00BF0     .WORD   9, 9            ;
0009  0009  00BF4     .WORD   9, 9            ;
0009  0009  00BF8     .WORD   9, 9            ;
0009  0009  00BFC     .WORD   9, 9            ;
0009  0009  00C00     .WORD   9, 9            ;
0009  0009  00C04     .WORD   9, 9            ;
0009  0009  00C08     .WORD   9, 9            ;
0009  0009  00C0C     .WORD   9, 9            ;
0009  0009  00C10     .WORD   9, 9            ;
0009  0009  00C14     .WORD   9, 9            ;
0009  0009  00C18     .WORD   9, 9            ;
0009  0009  00C1C     .WORD   9, 9            ;
0004  0003  00C20     .WORD   3, 4            ;
0005  0004  00C24     .WORD   4, 5            ;
0004  0005  00C28     .WORD   5, 4            ;
0009  0009  00C2C     .WORD   9, 9            ;
0007  0007  00C30     .WORD   7, 7            ;
0009  0009  00C34     .WORD   9, 9            ;
0009  0009  00C38     .WORD   9, 9            ;
0009  0009  00C3C     .WORD   9, 9            ;
0009  0009  00C40     .WORD   9, 9            ;
0009  0009  00C44     .WORD   9, 9            ;
0009  0009  00C48     .WORD   9, 9            ;
0009  0009  00C4C     .WORD   9, 9            ;
0009  0009  00C50     .WORD   9, 9            ;
0009  0009  00C54     .WORD   9, 9            ;
0009  0009  00C58     .WORD   9, 9            ;
0009  0009  00C5C     .WORD   9, 9            ;
0009  0009  00C60     .WORD   9, 9            ;
0009  0009  00C64     .WORD   9, 9            ;
0009  0009  00C68     .WORD   9, 9            ;
0009  0009  00C6C     .WORD   9, 9            ;
```

```
0009  0009  00C70     .WORD   9, 9            ;
0009  0009  00C74     .WORD   9, 9            ;
0009  0009  00C78     .WORD   9, 9            ;
0009  0009  00C7C     .WORD   9, 9            ;
0009  0009  00C80     .WORD   9, 9            ;
0009  0009  00C84     .WORD   9, 9            ;
0009  0009  00C88     .WORD   9, 9            ;
0009  0009  00C8C     .WORD   9, 9            ;
0009  0009  00C90     .WORD   9, 9            ;
0009  0009  00C94     .WORD   9, 9            ;
0009  0009  00C98     .WORD   9, 9            ;
0009  0009  00C9C     .WORD   9, 9            ;
0009  0009  00CA0     .WORD   9, 9            ;
0009  0009  00CA4     .WORD   9, 9            ;
0009  0009  00CA8     .WORD   9, 9            ;
0009  0009  00CAC     .WORD   9, 9            ;
0009  0009  00CB0     .WORD   9, 9            ;
0009  0009  00CB4     .WORD   9, 9            ;
0009  0009  00CB8     .WORD   9, 9            ;
0009  0009  00CBC     .WORD   9, 9            ;
0009  0009  00CC0     .WORD   9, 9            ;
0009  0009  00CC4     .WORD   9, 9            ;
0009  0009  00CC8     .WORD   9, 9            ;
0009  0009  00CCC     .WORD   9, 9            ;
0009  0009  00CD0     .WORD   9, 9            ;
0009  0009  00CD4     .WORD   9, 9            ;
0009  0009  00CD8     .WORD   9, 9            ;
0009  0009  00CDC     .WORD   9, 9            ;
0009  0009  00CE0     .WORD   9, 9            ;
0009  0009  00CE4     .WORD   9, 9            ;
0009  0009  00CE8     .WORD   9, 9            ;
0009  0009  00CEC     .WORD   9, 9            ;
0009  0009  00CF0     .WORD   9, 9            ;
0009  0009  00CF4     .WORD   9, 9            ;
0009  0009  00CF8     .WORD   9, 9            ;
0009  0009  00CFC     .WORD   9, 9            ;
0009  0009  00D00     .WORD   9, 9            ;
0009  0009  00D04     .WORD   9, 9            ;
0009  0009  00D08     .WORD   9, 9            ;
0009  0009  00D0C     .WORD   9, 9            ;
0009  0009  00D10     .WORD   9, 9            ;
0009  0009  00D14     .WORD   9, 9            ;
0009  0009  00D18     .WORD   9, 9            ;
0009  0009  00D1C     .WORD   9, 9            ;
0009  0009  00D20     .WORD   9, 9            ;
0009  0009  00D24     .WORD   9, 9            ;
0009  0009  00D28     .WORD   9, 9            ;
0009  0009  00D2C     .WORD   9, 9            ;
0009  0009  00D30     .WORD   9, 9            ;
0009  0009  00D34     .WORD   9, 9            ;
0009  0009  00D38     .WORD   9, 9            ;
0009  0009  00D3C     .WORD   9, 9            ;
0009  0009  00D40     .WORD   9, 9            ;
0009  0009  00D44     .WORD   9, 9            ;
0009  0009  00D48     .WORD   9, 9            ;
```

```
0009  0009  00D4C     .WORD   9, 9            ;
0009  0009  00D50     .WORD   9, 9            ;
0009  0009  00D54     .WORD   9, 9            ;
0009  0009  00D58     .WORD   9, 9            ;
0009  0009  00D5C     .WORD   9, 9            ;
0009  0009  00D60     .WORD   9, 9            ;
0009  0009  00D64     .WORD   9, 9            ;
0009  0009  00D68     .WORD   9, 9            ;
0009  0009  00D6C     .WORD   9, 9            ;
0009  0009  00D70     .WORD   9, 9            ;
0009  0009  00D74     .WORD   9, 9            ;
0009  0009  00D78     .WORD   9, 9            ;
0009  0009  00D7C     .WORD   9, 9            ;
0009  0009  00D80     .WORD   9, 9            ;
0009  0009  00D84     .WORD   9, 9            ;
0009  0009  00D88     .WORD   9, 9            ;
0009  0009  00D8C     .WORD   9, 9            ;
0009  0009  00D90     .WORD   9, 9            ;
0009  0009  00D94     .WORD   9, 9            ;
0009  0009  00D98     .WORD   9, 9            ;
0009  0009  00D9C     .WORD   9, 9            ;
0009  0009  00DA0     .WORD   9, 9            ;
0009  0009  00DA4     .WORD   9, 9            ;
0009  0009  00DA8     .WORD   9, 9            ;
0009  0009  00DAC     .WORD   9, 9            ;
0009  0009  00DB0     .WORD   9, 9            ;
0009  0009  00DB4     .WORD   9, 9            ;
0009  0009  00DB8     .WORD   9, 9            ;
0009  0009  00DBC     .WORD   9, 9            ;
0009  0009  00DC0     .WORD   9, 9            ;
0009  0009  00DC4     .WORD   9, 9            ;
0009  0009  00DC8     .WORD   9, 9            ;
0009  0009  00DCC     .WORD   9, 9            ;
0009  0009  00DD0     .WORD   9, 9            ;
0009  0009  00DD4     .WORD   9, 9            ;
0009  0009  00DD8     .WORD   9, 9            ;
0009  0009  00DDC     .WORD   9, 9            ;
0009  0009  00DE0     .WORD   9, 9            ;
0009  0009  00DE4     .WORD   9, 9            ;
0009  0009  00DE8     .WORD   9, 9            ;
0009  0009  00DEC     .WORD   9, 9            ;
0009  0009  00DF0     .WORD   9, 9            ;
0009  0009  00DF4     .WORD   9, 9            ;
0009  0009  00DF8     .WORD   9, 9            ;
0009  0009  00DFC     .WORD   9, 9            ;
0009  0009  00E00     .WORD   9, 9            ;
0009  0009  00E04     .WORD   9, 9            ;
0009  0009  00E08     .WORD   9, 9            ;
0009  0009  00E0C     .WORD   9, 9            ;
0009  0009  00E10     .WORD   9, 9            ;
0009  0009  00E14     .WORD   9, 9            ;
0009  0009  00E18     .WORD   9, 9            ;
0009  0009  00E1C     .WORD   9, 9            ;
0009  0009  00E20     .WORD   9, 9            ;
0009  0009  00E24     .WORD   9, 9            ;
```

```
0009  0009  00E28     .WORD   9, 9            ;
0009  0009  00E2C     .WORD   9, 9            ;
0009  0009  00E30     .WORD   9, 9            ;
0009  0009  00E34     .WORD   9, 9            ;
0009  0009  00E38     .WORD   9, 9            ;
0009  0009  00E3C     .WORD   9, 9            ;
0009  0009  00E40     .WORD   9, 9            ;
0009  0009  00E44     .WORD   9, 9            ;
0009  0009  00E48     .WORD   9, 9            ;
0009  0009  00E4C     .WORD   9, 9            ;
0009  0009  00E50     .WORD   9, 9            ;
0009  0009  00E54     .WORD   9, 9            ;
0009  0009  00E58     .WORD   9, 9            ;
0009  0009  00E5C     .WORD   9, 9            ;
0009  0009  00E60     .WORD   9, 9            ;
0009  0009  00E64     .WORD   9, 9            ;
0009  0009  00E68     .WORD   9, 9            ;
0009  0009  00E6C     .WORD   9, 9            ;
0009  0009  00E70     .WORD   9, 9            ;
0009  0009  00E74     .WORD   9, 9            ;
0009  0009  00E78     .WORD   9, 9            ;
0009  0009  00E7C     .WORD   9, 9            ;
0009  0009  00E80     .WORD   9, 9            ;
0009  0009  00E84     .WORD   9, 9            ;
0009  0009  00E88     .WORD   9, 9            ;
0009  0009  00E8C     .WORD   9, 9            ;
0009  0009  00E90     .WORD   9, 9            ;
0009  0009  00E94     .WORD   9, 9            ;
0009  0009  00E98     .WORD   9, 9            ;
0009  0009  00E9C     .WORD   9, 9            ;
0009  0009  00EA0     .WORD   9, 9            ;
0009  0009  00EA4     .WORD   9, 9            ;
0009  0009  00EA8     .WORD   9, 9            ;
0009  0009  00EAC     .WORD   9, 9            ;
0009  0009  00EB0     .WORD   9, 9            ;
0009  0009  00EB4     .WORD   9, 9            ;
0009  0009  00EB8     .WORD   9, 9            ;
0009  0009  00EBC     .WORD   9, 9            ;
0009  0009  00EC0     .WORD   9, 9            ;
0009  0009  00EC4     .WORD   9, 9            ;
0009  0009  00EC8     .WORD   9, 9            ;
0009  0009  00ECC     .WORD   9, 9            ;
0009  0009  00ED0     .WORD   9, 9            ;
0009  0009  00ED4     .WORD   9, 9            ;
0009  0009  00ED8     .WORD   9, 9            ;
0009  0009  00EDC     .WORD   9, 9            ;
0009  0009  00EE0     .WORD   9, 9            ;
0009  0009  00EE4     .WORD   9, 9            ;
0009  0009  00EE8     .WORD   9, 9            ;
0009  0009  00EEC     .WORD   9, 9            ;
0009  0009  00EF0     .WORD   9, 9            ;
0009  0009  00EF4     .WORD   9, 9            ;
0009  0009  00EF8     .WORD   9, 9            ;
0009  0009  00EFC     .WORD   9, 9            ;
0009  0009  00F00     .WORD   9, 9            ;
```

```
0009  0009  00F04     .WORD   9, 9            ;
0009  0009  00F08     .WORD   9, 9            ;
0009  0009  00F0C     .WORD   9, 9            ;
0009  0009  00F10     .WORD   9, 9            ;
0009  0009  00F14     .WORD   9, 9            ;
0009  0009  00F18     .WORD   9, 9            ;
0009  0009  00F1C     .WORD   9, 9            ;
0009  0009  00F20     .WORD   9, 9            ;
0009  0009  00F24     .WORD   9, 9            ;
0009  0009  00F28     .WORD   9, 9            ;
0009  0009  00F2C     .WORD   9, 9            ;
0009  0009  00F30     .WORD   9, 9            ;
0009  0009  00F34     .WORD   9, 9            ;
0009  0009  00F38     .WORD   9, 9            ;
0009  0009  00F3C     .WORD   9, 9            ;
0009  0009  00F40     .WORD   9, 9            ;
0009  0009  00F44     .WORD   9, 9            ;
0009  0009  00F48     .WORD   9, 9            ;
0009  0009  00F4C     .WORD   9, 9            ;
0009  0009  00F50     .WORD   9, 9            ;
0009  0009  00F54     .WORD   9, 9            ;
0009  0009  00F58     .WORD   9, 9            ;
0009  0009  00F5C     .WORD   9, 9            ;
0009  0009  00F60     .WORD   9, 9            ;
0009  0009  00F64     .WORD   9, 9            ;
0009  0009  00F68     .WORD   9, 9            ;
0009  0009  00F6C     .WORD   9, 9            ;
0009  0009  00F70     .WORD   9, 9            ;
0009  0009  00F74     .WORD   9, 9            ;
0009  0009  00F78     .WORD   9, 9            ;
0009  0009  00F7C     .WORD   9, 9            ;
0009  0009  00F80     .WORD   9, 9            ;
0009  0009  00F84     .WORD   9, 9            ;
0009  0009  00F88     .WORD   9, 9            ;
0009  0009  00F8C     .WORD   9, 9            ;
0009  0009  00F90     .WORD   9, 9            ;
0009  0009  00F94     .WORD   9, 9            ;
0009  0009  00F98     .WORD   9, 9            ;
0009  0009  00F9C     .WORD   9, 9            ;
0009  0009  00FA0     .WORD   9, 9            ;
0009  0009  00FA4     .WORD   9, 9            ;
0009  0009  00FA8     .WORD   9, 9            ;
0009  0009  00FAC     .WORD   9, 9            ;
0009  0009  00FB0     .WORD   9, 9            ;
0009  0009  00FB4     .WORD   9, 9            ;
0009  0009  00FB8     .WORD   9, 9            ;
0009  0009  00FBC     .WORD   9, 9            ;
0009  0009  00FC0     .WORD   9, 9            ;
0009  0009  00FC4     .WORD   9, 9            ;
0009  0009  00FC8     .WORD   9, 9            ;
0009  0009  00FCC     .WORD   9, 9            ;
0009  0009  00FD0     .WORD   9, 9            ;
0009  0009  00FD4     .WORD   9, 9            ;
0009  0009  00FD8     .WORD   9, 9            ;
0009  0009  00FDC     .WORD   9, 9            ;
```

```
0009  0009  00FE0     .WORD   9, 9            ;
0009  0009  00FE4     .WORD   9, 9            ;
0009  0009  00FE8     .WORD   9, 9            ;
0009  0009  00FEC     .WORD   9, 9            ;
0009  0009  00FF0     .WORD   9, 9            ;
0009  0009  00FF4     .WORD   9, 9            ;
0009  0009  00FF8     .WORD   9, 9            ;
0009  0009  00FFC     .WORD   9, 9            ;
0009  0009  01000     .WORD   9, 9            ;
0009  0009  01004     .WORD   9, 9            ;
0009  0009  01008     .WORD   9, 9            ;
0009  0009  0100C     .WORD   9, 9            ;
0009  0009  01010     .WORD   9, 9            ;
0009  0009  01014     .WORD   9, 9            ;
0009  0009  01018     .WORD   9, 9            ;
0009  0009  0101C     .WORD   9, 9            ;
0009  0009  01020     .WORD   9, 9            ;
0009  0009  01024     .WORD   9, 9            ;
0009  0009  01028     .WORD   9, 9            ;
0009  0009  0102C     .WORD   9, 9            ;
0009  0009  01030     .WORD   9, 9            ;
0009  0009  01034     .WORD   9, 9            ;
0009  0009  01038     .WORD   9, 9            ;
0009  0009  0103C     .WORD   9, 9            ;
0009  0009  01040     .WORD   9, 9            ;
0009  0009  01044     .WORD   9, 9            ;
0009  0009  01048     .WORD   9, 9            ;
0009  0009  0104C     .WORD   9, 9            ;
0009  0009  01050     .WORD   9, 9            ;
0009  0009  01054     .WORD   9, 9            ;
0009  0009  01058     .WORD   9, 9            ;
0009  0009  0105C     .WORD   9, 9            ;
0009  0009  01060     .WORD   9, 9            ;
0009  0009  01064     .WORD   9, 9            ;
0009  0009  01068     .WORD   9, 9            ;
0009  0009  0106C     .WORD   9, 9            ;
0009  0009  01070     .WORD   9, 9            ;
0009  0009  01074     .WORD   9, 9            ;
0009  0009  01078     .WORD   9, 9            ;
0009  0009  0107C     .WORD   9, 9            ;
0009  0009  01080     .WORD   9, 9            ;
0009  0009  01084     .WORD   9, 9            ;
0009  0009  01088     .WORD   9, 9            ;
0009  0009  0108C     .WORD   9, 9            ;
0009  0009  01090     .WORD   9, 9            ;
0009  0009  01094     .WORD   9, 9            ;
0009  0009  01098     .WORD   9, 9            ;
0009  0009  0109C     .WORD   9, 9            ;
0009  0009  010A0     .WORD   9, 9            ;
0009  0009  010A4     .WORD   9, 9            ;
0009  0009  010A8     .WORD   9, 9            ;
0009  0009  010AC     .WORD   9, 9            ;
0009  0009  010B0     .WORD   9, 9            ;
0009  0009  010B4     .WORD   9, 9            ;
0009  0009  010B8     .WORD   9, 9            ;
```

```
0009  0009  010BC     .WORD   9, 9            ;
0009  0009  010C0     .WORD   9, 9            ;
0009  0009  010C4     .WORD   9, 9            ;
0009  0009  010C8     .WORD   9, 9            ;
0009  0009  010CC     .WORD   9, 9            ;
0009  0009  010D0     .WORD   9, 9            ;
0009  0009  010D4     .WORD   9, 9            ;
0009  0009  010D8     .WORD   9, 9            ;
0009  0009  010DC     .WORD   9, 9            ;
0009  0009  010E0     .WORD   9, 9            ;
0009  0009  010E4     .WORD   9, 9            ;
0009  0009  010E8     .WORD   9, 9            ;
0009  0009  010EC     .WORD   9, 9            ;
0009  0009  010F0     .WORD   9, 9            ;
0009  0009  010F4     .WORD   9, 9            ;
0009  0009  010F8     .WORD   9, 9            ;
0009  0009  010FC     .WORD   9, 9            ;
0009  0009  01100     .WORD   9, 9            ;
0009  0009  01104     .WORD   9, 9            ;
0009  0009  01108     .WORD   9, 9            ;
0009  0009  0110C     .WORD   9, 9            ;
0009  0009  01110     .WORD   9, 9            ;
0009  0009  01114     .WORD   9, 9            ;
0009  0009  01118     .WORD   9, 9            ;
0009  0009  0111C     .WORD   9, 9            ;
0009  0009  01120     .WORD   9, 9            ;
0009  0009  01124     .WORD   9, 9            ;
0009  0009  01128     .WORD   9, 9            ;
0009  0009  0112C     .WORD   9, 9            ;
0009  0009  01130     .WORD   9, 9            ;
0009  0009  01134     .WORD   9, 9            ;
0009  0009  01138     .WORD   9, 9            ;
0009  0009  0113C     .WORD   9, 9            ;
0009  0009  01140     .WORD   9, 9            ;
0009  0009  01144     .WORD   9, 9            ;
0009  0009  01148     .WORD   9, 9            ;
0009  0009  0114C     .WORD   9, 9            ;
0009  0009  01150     .WORD   9, 9            ;
0009  0009  01154     .WORD   9, 9            ;
0009  0009  01158     .WORD   9, 9            ;
0009  0009  0115C     .WORD   9, 9            ;
0009  0009  01160     .WORD   9, 9            ;
0009  0009  01164     .WORD   9, 9            ;
0009  0009  01168     .WORD   9, 9            ;
0009  0009  0116C     .WORD   9, 9            ;
0009  0009  01170     .WORD   9, 9            ;
0009  0009  01174     .WORD   9, 9            ;
0009  0009  01178     .WORD   9, 9            ;
0009  0009  0117C     .WORD   9, 9            ;
0009  0009  01180     .WORD   9, 9            ;
0009  0009  01184     .WORD   9, 9            ;
0009  0009  01188     .WORD   9, 9            ;
0009  0009  0118C     .WORD   9, 9            ;
0009  0009  01190     .WORD   9, 9            ;
0009  0009  01194     .WORD   9, 9            ;
```

```
0009  0009  01198     .WORD   9, 9            ;
0009  0009  0119C     .WORD   9, 9            ;
0009  0009  011A0     .WORD   9, 9            ;
0009  0009  011A4     .WORD   9, 9            ;
0009  0009  011A8     .WORD   9, 9            ;
0009  0009  011AC     .WORD   9, 9            ;
0009  0009  011B0     .WORD   9, 9            ;
0009  0009  011B4     .WORD   9, 9            ;
0009  0009  011B8     .WORD   9, 9            ;
0009  0009  011BC     .WORD   9, 9            ;
0009  0009  011C0     .WORD   9, 9            ;
0009  0009  011C4     .WORD   9, 9            ;
0009  0009  011C8     .WORD   9, 9            ;
0009  0009  011CC     .WORD   9, 9            ;
0009  0009  011D0     .WORD   9, 9            ;
0009  0009  011D4     .WORD   9, 9            ;
0009  0009  011D8     .WORD   9, 9            ;
0015  0015  011DC P.AAD:  .WORD   21, 21          ;
0015  0015  011E0     .WORD   21, 21           ;
0015  0015  011E4     .WORD   21, 21           ;
0015  0015  011E8     .WORD   21, 21           ;
0015  0015  011EC     .WORD   21, 21           ;
0015  0015  011F0     .WORD   21, 21           ;
0015  0015  011F4     .WORD   21, 21           ;
0015  0015  011F8     .WORD   21, 21           ;
0015  0015  011FC     .WORD   21, 21           ;
0015  0015  01200     .WORD   21, 21           ;
0015  0015  01204     .WORD   21, 21           ;
0015  0015  01208     .WORD   21, 21           ;
0015  0015  0120C     .WORD   21, 21           ;
0015  0015  01210     .WORD   21, 21           ;
000E  0000  01214     .WORD   0, 14            ;
0015  0015  01218     .WORD   21, 21           ;
0015  0015  0121C     .WORD   21, 21           ;
0011  0000  01220     .WORD   0, 17            ;
0012  0000  01224     .WORD   0, 18            ;
0013  0000  01228     .WORD   0, 19            ;
0014  0000  0122C     .WORD   0, 20            ;
0015  0015  01230     .WORD   21, 21           ;
0015  0015  01234     .WORD   21, 21           ;
0015  0015  01238     .WORD   21, 21           ;
0015  0015  0123C     .WORD   21, 21           ;
0004  0003  01240     .WORD   3, 4            ;
0005  0004  01244     .WORD   4, 5            ;
0006  0005  01248     .WORD   5, 6            ;
0005  0006  0124C     .WORD   6, 5            ;
0009  0007  01250     .WORD   7, 9            ;
0015  0015  01254     .WORD   21, 21           ;
000A  0009  01258     .WORD   9, 10            ;
000B  000A  0125C     .WORD   10, 11           ;
000C  000B  01260     .WORD   11, 12           ;
000D  000C  01264     .WORD   12, 13           ;
000E  000D  01268     .WORD   13, 14           ;
0004  0001  0126C     .WORD   1, 4            ;
0015  0015  01270     .WORD   21, 21           ;
```

```
      0010  0010  01274     .WORD   16, 16          ;
      0004  0001  01278     .WORD   1, 4           ;
      0004  0001  0127C     .WORD   1, 4           ;
      0010  0001  01280     .WORD   1, 16           ;
      0004  0001  01284     .WORD   1, 4           ;
      0015  0015  01288     .WORD   21, 21          ;
      0015  0015  0128C     .WORD   21, 21          ;
      0015  0015  01290     .WORD   21, 21          ;
      0015  0015  01294     .WORD   21, 21          ;
      0015  0015  01298     .WORD   21, 21          ;
      0015  0015  0129C     .WORD   21, 21          ;
      0015  0015  012A0     .WORD   21, 21          ;
      0015  0015  012A4     .WORD   21, 21          ;
      0015  0015  012A8     .WORD   21, 21          ;
      0015  0015  012AC     .WORD   21, 21          ;
      0015  0015  012B0     .WORD   21, 21          ;
      0015  0015  012B4     .WORD   21, 21          ;
      0015  0015  012B8     .WORD   21, 21          ;
      0015  0015  012BC     .WORD   21, 21          ;
      0015  0015  012C0     .WORD   21, 21          ;
      0013  0013  012C4     .WORD   19, 19          ;
      0015  0015  012C8     .WORD   21, 21          ;
      0015  0015  012CC     .WORD   21, 21          ;
      0015  0015  012D0     .WORD   21, 21          ;
      0015  0015  012D4     .WORD   21, 21          ;
      0013  0000  012D8     .WORD   0, 19           ;
      0014  0000  012DC     .WORD   0, 20           ;
      0015  0015  012E0     .WORD   21, 21          ;
      0015  0015  012E4     .WORD   21, 21          ;
      0015  0015  012E8     .WORD   21, 21          ;
      0015  0015  012EC     .WORD   21, 21          ;
      0015  0015  012F0     .WORD   21, 21          ;
      0015  0015  012F4     .WORD   21, 21          ;
      0015  0015  012F8     .WORD   21, 21          ;
      0015  0015  012FC     .WORD   21, 21          ;
      0015  0015  01300     .WORD   21, 21          ;
      0015  0015  01304     .WORD   21, 21          ;
      0015  0015  01308     .WORD   21, 21          ;
      0011  0011  0130C     .WORD   17, 17          ;
      0012  0012  01310     .WORD   18, 18          ;
      0012  0012  01314     .WORD   18, 18          ;
      0012  0012  01318     .WORD   18, 18          ;
      0012  0012  0131C     .WORD   18, 18          ;
      0015  0015  01320     .WORD   21, 21          ;
      0015  0015  01324     .WORD   21, 21          ;
      0011  0000  01328     .WORD   0, 17           ;
      0015  0015  0132C     .WORD   21, 21          ;
      0013  0000  01330     .WORD   0, 19           ;
      0014  0000  01334     .WORD   0, 20           ;
      0015  0015  01338     .WORD   21, 21          ;
      0015  0015  0133C     .WORD   21, 21          ;
      0015  0015  01340     .WORD   21, 21          ;
      0015  0015  01344     .WORD   21, 21          ;
      0015  0015  01348     .WORD   21, 21          ;
      0015  0015  0134C     .WORD   21, 21          ;
```

```
      0015  0015  01350     .WORD   21, 21          ;
      0015  0015  01354     .WORD   21, 21          ;
      0015  0015  01358     .WORD   21, 21          ;
      0015  0015  0135C     .WORD   21, 21          ;
      0015  0015  01360     .WORD   21, 21          ;
      0015  0015  01364     .WORD   21, 21          ;
      0015  0015  01368     .WORD   21, 21          ;
      0015  0015  0136C     .WORD   21, 21          ;
      0015  0015  01370     .WORD   21, 21          ;
      0015  0015  01374     .WORD   21, 21          ;
      0015  0015  01378     .WORD   21, 21          ;
      0015  0015  0137C     .WORD   21, 21          ;
      0015  0015  01380     .WORD   21, 21          ;
      0015  0015  01384     .WORD   21, 21          ;
      0015  0015  01388     .WORD   21, 21          ;
      0015  0015  0138C     .WORD   21, 21          ;
      0015  0015  01390     .WORD   21, 21          ;
      0015  0015  01394     .WORD   21, 21          ;
      0015  0015  01398     .WORD   21, 21          ;
      0015  0015  0139C     .WORD   21, 21          ;
      0015  0015  013A0     .WORD   21, 21          ;
      0015  0015  013A4     .WORD   21, 21          ;
      0015  0015  013A8     .WORD   21, 21          ;
      0015  0015  013AC     .WORD   21, 21          ;
      0015  0015  013B0     .WORD   21, 21          ;
      0015  0015  013B4     .WORD   21, 21          ;
      0015  0015  013B8     .WORD   21, 21          ;
      0015  0015  013BC     .WORD   21, 21          ;
      0015  0015  013C0     .WORD   21, 21          ;
      0015  0015  013C4     .WORD   21, 21          ;
      0015  0015  013C8     .WORD   21, 21          ;
      0013  0013  013CC     .WORD   19, 19          ;
      0015  0015  013D0     .WORD   21, 21          ;
      0015  0015  013D4     .WORD   21, 21          ;
      0015  0015  013D8     .WORD   21, 21          ;
      0015  0015  013DC     .WORD   21, 21          ;
      0013  0000  013E0     .WORD   0, 19           ;
      0014  0000  013E4     .WORD   0, 20           ;
      0015  0015  013E8     .WORD   21, 21          ;
      0015  0015  013EC     .WORD   21, 21          ;
      0015  0015  013F0     .WORD   21, 21          ;
      0015  0015  013F4     .WORD   21, 21          ;
      0015  0015  013F8     .WORD   21, 21          ;
      0015  0015  013FC     .WORD   21, 21          ;
      0015  0015  01400     .WORD   21, 21          ;
      0015  0015  01404     .WORD   21, 21          ;
      0015  0015  01408     .WORD   21, 21          ;
      0015  0015  0140C     .WORD   21, 21          ;
      0015  0015  01410     .WORD   21, 21          ;
      0015  0015  01414     .WORD   21, 21          ;
      0015  0015  01418     .WORD   21, 21          ;
      0015  0015  0141C     .WORD   21, 21          ;
      0015  0015  01420     .WORD   21, 21          ;
      0013  0013  01424     .WORD   19, 19          ;
      0015  0015  01428     .WORD   21, 21          ;
```

```
      0015  0015  0142C     .WORD   21, 21          ;
      0015  0015  01430     .WORD   21, 21          ;
      0015  0015  01434     .WORD   21, 21          ;
      0013  0000  01438     .WORD   0, 19           ;
      0014  0000  0143C     .WORD   0, 20           ;
      0015  0015  01440     .WORD   21, 21          ;
      0015  0015  01444     .WORD   21, 21          ;
      0015  0015  01448     .WORD   21, 21          ;
      0015  0015  0144C     .WORD   21, 21          ;
      0015  0015  01450     .WORD   21, 21          ;
      0015  0015  01454     .WORD   21, 21          ;
      0015  0015  01458     .WORD   21, 21          ;
      0015  0015  0145C     .WORD   21, 21          ;
      0015  0015  01460     .WORD   21, 21          ;
      0015  0015  01464     .WORD   21, 21          ;
      0015  0015  01468     .WORD   21, 21          ;
      0015  0015  0146C     .WORD   21, 21          ;
      0015  0015  01470     .WORD   21, 21          ;
      0015  0015  01474     .WORD   21, 21          ;
      0015  0015  01478     .WORD   21, 21          ;
      0013  0013  0147C     .WORD   19, 19          ;
      0015  0015  01480     .WORD   21, 21          ;
      0015  0015  01484     .WORD   21, 21          ;
      0015  0015  01488     .WORD   21, 21          ;
      0015  0015  0148C     .WORD   21, 21          ;
      0013  0000  01490     .WORD   0, 19           ;
      0014  0000  01494     .WORD   0, 20           ;
      0015  0015  01498     .WORD   21, 21          ;
      0015  0015  0149C     .WORD   21, 21          ;
      0015  0015  014A0     .WORD   21, 21          ;
      0015  0015  014A4     .WORD   21, 21          ;
      0015  0015  014A8     .WORD   21, 21          ;
      0015  0015  014AC     .WORD   21, 21          ;
      0015  0015  014B0     .WORD   21, 21          ;
      0015  0015  014B4     .WORD   21, 21          ;
      0015  0015  014B8     .WORD   21, 21          ;
      0015  0015  014BC     .WORD   21, 21          ;
      0015  0015  014C0     .WORD   21, 21          ;
      0015  0015  014C4     .WORD   21, 21          ;
      0015  0015  014C8     .WORD   21, 21          ;
      0015  0015  014CC     .WORD   21, 21          ;
      0015  0015  014D0     .WORD   21, 21          ;
      0013  0013  014D4     .WORD   19, 19          ;
      0015  0015  014D8     .WORD   21, 21          ;
      0015  0015  014DC     .WORD   21, 21          ;
      0015  0015  014E0     .WORD   21, 21          ;
      0015  0015  014E4     .WORD   21, 21          ;
      0013  0000  014E8     .WORD   0, 19           ;
      0014  0000  014EC     .WORD   0, 20           ;
      0015  0015  014F0     .WORD   21, 21          ;
      0015  0015  014F4     .WORD   21, 21          ;
      0015  0015  014F8     .WORD   21, 21          ;
      0015  0015  014FC     .WORD   21, 21          ;
      0015  0015  01500     .WORD   21, 21          ;
      0015  0015  01504     .WORD   21, 21          ;
```

```
      0015  0015  01508     .WORD   21, 21           ;
      0015  0015  0150C     .WORD   21, 21           ;
      0015  0015  01510     .WORD   21, 21           ;
      0015  0015  01514     .WORD   21, 21           ;
      0015  0015  01518     .WORD   21, 21           ;
      0015  0015  0151C     .WORD   21, 21           ;
      0015  0015  01520     .WORD   21, 21           ;
      0015  0015  01524     .WORD   21, 21           ;
      0015  0015  01528     .WORD   21, 21           ;
      0013  0013  0152C     .WORD   19, 19           ;
      0015  0015  01530     .WORD   21, 21           ;
      0015  0015  01534     .WORD   21, 21           ;
      0015  0015  01538     .WORD   21, 21           ;
      0015  0015  0153C     .WORD   21, 21           ;
      0013  0000  01540     .WORD   0, 19            ;
      0014  0000  01544     .WORD   0, 20            ;
      0015  0015  01548     .WORD   21, 21           ;
      0015  0015  0154C     .WORD   21, 21           ;
      0015  0015  01550     .WORD   21, 21           ;
      0015  0015  01554     .WORD   21, 21           ;
      0015  0015  01558     .WORD   21, 21           ;
      0015  0015  0155C     .WORD   21, 21           ;
      0015  0015  01560     .WORD   21, 21           ;
      0015  0015  01564     .WORD   21, 21           ;
      0015  0015  01568     .WORD   21, 21           ;
      0015  0015  0156C     .WORD   21, 21           ;
      0015  0015  01570     .WORD   21, 21           ;
      0015  0015  01574     .WORD   21, 21           ;
      0015  0015  01578     .WORD   21, 21           ;
      0015  0015  0157C     .WORD   21, 21           ;
      0015  0015  01580     .WORD   21, 21           ;
      0013  0013  01584     .WORD   19, 19           ;
      0015  0015  01588     .WORD   21, 21           ;
      0015  0015  0158C     .WORD   21, 21           ;
      0015  0015  01590     .WORD   21, 21           ;
      0015  0015  01594     .WORD   21, 21           ;
      0013  0000  01598     .WORD   0, 19            ;
      0014  0000  0159C     .WORD   0, 20            ;
      0015  0015  015A0     .WORD   21, 21           ;
      0015  0015  015A4     .WORD   21, 21           ;
      0015  0015  015A8     .WORD   21, 21           ;
      0015  0015  015AC     .WORD   21, 21           ;
      0015  0015  015B0     .WORD   21, 21           ;
      0015  0015  015B4     .WORD   21, 21           ;
      0015  0015  015B8     .WORD   21, 21           ;
      0015  0015  015BC     .WORD   21, 21           ;
      0015  0015  015C0     .WORD   21, 21           ;
      0015  0015  015C4     .WORD   21, 21           ;
      0015  0015  015C8     .WORD   21, 21           ;
      0015  0015  015CC     .WORD   21, 21           ;
      0015  0015  015D0     .WORD   21, 21           ;
      0015  0015  015D4     .WORD   21, 21           ;
      0015  0015  015D8     .WORD   21, 21           ;
      0013  0013  015DC     .WORD   19, 19           ;
      0015  0015  015E0     .WOPD   21, 21           ;
```

```
0015  0015  015E4     .WORD   21, 21          ;
0015  0015  015E8     .WORD   21, 21          ;
0015  0015  015EC     .WORD   21, 21          ;
0015  0015  015F0     .WORD   21, 21          ;
0015  0015  015F4     .WORD   21, 21          ;
0015  0015  015F8     .WORD   21, 21          ;
0015  0015  015FC     .WORD   21, 21          ;
0015  0015  01600     .WORD   21, 21          ;
0015  0015  01604     .WORD   21, 21          ;
0015  0015  01608     .WORD   21, 21          ;
0015  0015  0160C     .WORD   21, 21          ;
0015  0015  01610     .WORD   21, 21          ;
0015  0015  01614     .WORD   21, 21          ;
0015  0015  01618     .WORD   21, 21          ;
0015  0015  0161C     .WORD   21, 21          ;
0015  0015  01620     .WORD   21, 21          ;
0015  0015  01624     .WORD   21, 21          ;
0015  0015  01628     .WORD   21, 21          ;
0015  0015  0162C     .WORD   21, 21          ;
0015  0015  01630     .WORD   21, 21          ;
0013  0013  01634     .WORD   19, 19          ;
0015  0015  01638     .WORD   21, 21          ;
0015  0015  0163C     .WORD   21, 21          ;
0015  0015  01640     .WORD   21, 21          ;
0015  0015  01644     .WORD   21, 21          ;
0013  0000  01648     .WORD   0, 19           ;
0014  0000  0164C     .WORD   0, 20           ;
0015  0015  01650     .WORD   21, 21          ;
0015  0015  01654     .WORD   21, 21          ;
0015  0015  01658     .WORD   21, 21          ;
0015  0015  0165C     .WORD   21, 21          ;
0015  0015  01660     .WORD   21, 21          ;
0015  0015  01664     .WORD   21, 21          ;
0015  0015  01668     .WORD   21, 21          ;
0015  0015  0166C     .WORD   21, 21          ;
0015  0015  01670     .WORD   21, 21          ;
0015  0015  01674     .WORD   21, 21          ;
0015  0015  01678     .WORD   21, 21          ;
0015  0015  0167C     .WORD   21, 21          ;
0015  0015  01680     .WORD   21, 21          ;
0015  0015  01684     .WORD   21, 21          ;
0015  0015  01688     .WORD   21, 21          ;
0013  0013  0168C     .WORD   19, 19          ;
0015  0015  01690     .WORD   21, 21          ;
0015  0015  01694     .WORD   21, 21          ;
0015  0015  01698     .WORD   21, 21          ;
0015  0015  0169C     .WORD   21, 21          ;
0013  0000  016A0     .WORD   0, 19           ;
0014  0000  016A4     .WORD   0, 20           ;
0015  0015  016A8     .WORD   21, 21          ;
0015  0015  016AC     .WORD   21, 21          ;
0015  0015  016B0     .WORD   21, 21          ;
0015  0015  016B4     .WORD   21, 21          ;
0015  0015  016B8     .WORD   21, 21          ;
0015  0015  016BC     .WORD   21, 21          ;
```

```
0015  0015  016C0     .WORD   21, 21          ;
0015  0015  016C4     .WORD   21, 21          ;
0015  0015  016C8     .WORD   21, 21          ;
0015  0015  016CC     .WORD   21, 21          ;
0015  0015  016D0     .WORD   21, 21          ;
0015  0015  016D4     .WORD   21, 21          ;
0015  0015  016D8     .WORD   21, 21          ;
0015  0015  016DC     .WORD   21, 21          ;
0015  0015  016E0     .WORD   21, 21          ;
000E  0000  016E4     .WORD   0, 14           ;
0015  0015  016E8     .WORD   21, 21          ;
0015  0015  016EC     .WORD   21, 21          ;
0015  0015  016F0     .WORD   21, 21          ;
0015  0015  016F4     .WORD   21, 21          ;
0013  0000  016F8     .WORD   0, 19           ;
0014  0000  016FC     .WORD   0, 20           ;
0015  0015  01700     .WORD   21, 21          ;
0015  0015  01704     .WORD   21, 21          ;
0015  0015  01708     .WORD   21, 21          ;
0015  0015  0170C     .WORD   21, 21          ;
0015  0015  01710     .WORD   21, 21          ;
0015  0015  01714     .WORD   21, 21          ;
0015  0015  01718     .WORD   21, 21          ;
0015  0015  0171C     .WORD   21, 21          ;
0015  0015  01720     .WORD   21, 21          ;
0015  0015  01724     .WORD   21, 21          ;
0015  0015  01728     .WORD   21, 21          ;
0015  0015  0172C     .WORD   21, 21          ;
0015  0015  01730     .WORD   21, 21          ;
0015  0015  01734     .WORD   21, 21          ;
0015  0015  01738     .WORD   21, 21          ;
000E  0000  0173C     .WORD   0, 14           ;
0015  0015  01740     .WORD   21, 21          ;
0015  0015  01744     .WORD   21, 21          ;
0015  0015  01748     .WORD   21, 21          ;
0015  0015  0174C     .WORD   21, 21          ;
0015  0015  01750     .WORD   21, 21          ;
0015  0015  01754     .WORD   21, 21          ;
0015  0015  01758     .WORD   21, 21          ;
0015  0015  0175C     .WORD   21, 21          ;
0015  0015  01760     .WORD   21, 21          ;
0015  0015  01764     .WORD   21, 21          ;
0015  0015  01768     .WORD   21, 21          ;
0015  0015  0176C     .WORD   21, 21          ;
0015  0015  01770     .WORD   21, 21          ;
0015  0015  01774     .WORD   21, 21          ;
0015  0015  01778     .WORD   21, 21          ;
0015  0015  0177C     .WORD   21, 21          ;
0015  0015  01780     .WORD   21, 21          ;
0015  0015  01784     .WORD   21, 21          ;
0015  0015  01788     .WORD   21, 21          ;
0015  0015  0178C     .WORD   21, 21          ;
0015  0015  01790     .WORD   21, 21          ;
000E  0000  01794     .WORD   0, 14           ;
0015  0015  01798     .WORD   21, 21          ;
```

```
0015  0015  0179C     .WORD   21, 21          ;
0015  0015  017A0     .WORD   21, 21          ;
0015  0015  017A4     .WORD   21, 21          ;
0013  0000  017A8     .WORD   0, 19           ;
0014  0000  017AC     .WORD   0, 20           ;
0015  0015  017B0     .WORD   21, 21          ;
0015  0015  017B4     .WORD   21, 21          ;
0015  0015  017B8     .WORD   21, 21          ;
0015  0015  017BC     .WORD   21, 21          ;
0015  0015  017C0     .WORD   21, 21          ;
0015  0015  017C4     .WORD   21, 21          ;
0015  0015  017C8     .WORD   21, 21          ;
0015  0015  017CC     .WORD   21, 21          ;
0015  0015  017D0     .WORD   21, 21          ;
0015  0015  017D4     .WORD   21, 21          ;
0015  0015  017D8     .WORD   21, 21          ;
0015  0015  017DC     .WORD   21, 21          ;
0015  0015  017E0     .WORD   21, 21          ;
0015  0015  017E4     .WORD   21, 21          ;
0015  0015  017E8     .WORD   21, 21          ;
000E  0000  017EC     .WORD   0, 14           ;
0015  0015  017F0     .WORD   21, 21          ;
0015  0015  017F4     .WORD   21, 21          ;
0015  0015  017F8     .WORD   21, 21          ;
0015  0015  017FC     .WORD   21, 21          ;
0015  0015  01800     .WORD   21, 21          ;
0015  0015  01804     .WORD   21, 21          ;
0015  0015  01808     .WORD   21, 21          ;
0015  0015  0180C     .WORD   21, 21          ;
0015  0015  01810     .WORD   21, 21          ;
0015  0015  01814     .WORD   21, 21          ;
0015  0015  01818     .WORD   21, 21          ;
0015  0015  0181C     .WORD   21, 21          ;
0015  0015  01820     .WORD   21, 21          ;
0015  0015  01824     .WORD   21, 21          ;
0015  0015  01828     .WORD   21, 21          ;
0015  0015  0182C     .WORD   21, 21          ;
0015  0015  01830     .WORD   21, 21          ;
0015  0015  01834     .WORD   21, 21          ;
0015  0015  01838     .WORD   21, 21          ;
0015  0015  0183C     .WORD   21, 21          ;
0015  0015  01840     .WORD   21, 21          ;
0013  0013  01844     .WORD   19, 19          ;
0015  0015  01848     .WORD   21, 21          ;
0015  0015  0184C     .WORD   21, 21          ;
0015  0015  01850     .WORD   21, 21          ;
0015  0015  01854     .WORD   21, 21          ;
0013  0000  01858     .WORD   0, 19           ;
0014  0000  0185C     .WORD   0, 20           ;
0015  0015  01860     .WORD   21, 21          ;
0015  0015  01864     .WORD   21, 21          ;
0015  0015  01868     .WORD   21, 21          ;
0015  0015  0186C     .WORD   21, 21          ;
0015  0015  01870     .WORD   21, 21          ;
0015  0015  01874     .WORD   21, 21          ;
```

```
      0015  0015  01878     .WORD   21, 21           ;
      0015  0015  0187C     .WORD   21, 21           ;
      0015  0015  01880     .WORD   21, 21           ;
      0015  0015  01884     .WORD   21, 21           ;
      0015  0015  01888     .WORD   21, 21           ;
      0015  0015  0188C     .WORD   21, 21           ;
      0015  0015  01890     .WORD   21, 21           ;
      0015  0015  01894     .WORD   21, 21           ;
      0015  0015  01898     .WORD   21, 21           ;
      0013  0013  0189C     .WORD   19, 19           ;
      0015  0015  018A0     .WORD   21, 21           ;
      0015  0015  018A4     .WORD   21, 21           ;
      0015  0015  018A8     .WORD   21, 21           ;
      0015  0015  018AC     .WORD   21, 21           ;
      0013  0000  018B0     .WORD   0, 19            ;
      0014  0000  018B4     .WORD   0, 20            ;
      0015  0015  018B8     .WORD   21, 21           ;
      0015  0015  018BC     .WORD   21, 21           ;
      0015  0015  018C0     .WORD   21, 21           ;
      0015  0015  018C4     .WORD   21, 21           ;
      0015  0015  018C8     .WORD   21, 21           ;
      0015  0015  018CC     .WORD   21, 21           ;
      0015  0015  018D0     .WORD   21, 21           ;
      0015  0015  018D4     .WORD   21, 21           ;
      0015  0015  018D8     .WORD   21, 21           ;
      0015  0015  018DC     .WORD   21, 21           ;
      0015  0015  018E0     .WORD   21, 21           ;
      0015  0015  018E4     .WORD   21, 21           ;
      0015  0015  018E8     .WORD   21, 21           ;
      0015  0015  018EC     .WORD   21, 21           ;
      0015  0015  018F0     .WORD   21, 21           ;
      0013  0013  018F4     .WORD   19, 19           ;
      0015  0015  018F8     .WORD   21, 21           ;
      0015  0015  018FC     .WORD   21, 21           ;
      0015  0015  01900     .WORD   21, 21           ;
      0015  0015  01904     .WORD   21, 21           ;
      0013  0000  01908     .WORD   0, 19            ;
      0014  0000  0190C     .WORD   0, 20            ;
      0015  0015  01910     .WORD   21, 21           ;
      0015  0015  01914     .WORD   21, 21           ;
      0015  0015  01918     .WORD   21, 21           ;
      0015  0015  0191C     .WORD   21, 21           ;
      0015  0015  01920     .WORD   21, 21           ;
      0015  0015  01924     .WORD   21, 21           ;
      0015  0015  01928     .WORD   21, 21           ;
      0015  0015  0192C     .WORD   21, 21           ;
      0015  0015  01930     .WORD   21, 21           ;
      0015  0015  01934     .WORD   21, 21           ;
      0015  0015  01938     .WORD   21, 21           ;
      0011  0011  0193C     .WORD   17, 17           ;
      0012  0012  01940     .WORD   18, 18           ;
      0012  0012  01944     .WORD   18, 18           ;
      0012  0012  01948     .WORD   18, 18           ;
      0012  0012  0194C     .WORD   18, 18           ;
      0015  0015  01950     .WORD   21, 21           ;
```

```
0015  0015  01954     .WORD   21, 21           ;
0011  0000  01958     .WORD   0, 17           ;
0012  0000  0195C     .WORD   0, 18           ;
0013  0000  01960     .WORD   0, 19           ;
0014  0000  01964     .WORD   0, 20           ;
0015  0015  01968     .WORD   21, 21          ;
0015  0015  0196C     .WORD   21, 21          ;
0015  0015  01970     .WORD   21, 21          ;
0015  0015  01974     .WORD   21, 21          ;
0015  0015  01978     .WORD   21, 21          ;
0015  0015  0197C     .WORD   21, 21          ;
0015  0015  01980     .WORD   21, 21          ;
0015  0015  01984     .WORD   21, 21          ;
0015  0015  01988     .WORD   21, 21          ;
0015  0015  0198C     .WORD   21, 21          ;
0015  0015  01990     .WORD   21, 21          ;
000A  0000  01994     .WORD   0, 10           ;
000B  0000  01998     .WORD   0, 11           ;
000C  0000  0199C     .WORD   0, 12           ;
000D  0000  019A0     .WORD   0, 13           ;
000E  0000  019A4     .WORD   0, 14           ;
0015  0015  019A8     .WORD   21, 21          ;
0015  0015  019AC     .WORD   21, 21          ;
0011  0000  019B0     .WORD   0, 17           ;
0012  0000  019B4     .WORD   0, 18           ;
0013  0000  019B8     .WORD   0, 19           ;
0014  0000  019BC     .WORD   0, 20           ;
0015  0015  019C0     .WORD   21, 21          ;
0015  0015  019C4     .WORD   21, 21          ;
0015  0015  019C8     .WORD   21, 21          ;
0015  0015  019CC     .WORD   21, 21          ;
0015  0015  019D0     .WORD   21, 21          ;
0015  0015  019D4     .WORD   21, 21          ;
0015  0015  019D8     .WORD   21, 21          ;
0015  0015  019DC     .WORD   21, 21          ;
0015  0015  019E0     .WORD   21, 21          ;
0015  0015  019E4     .WORD   21, 21          ;
0015  0015  019E8     .WORD   21, 21          ;
0015  0015  019EC     .WORD   21, 21          ;
0015  0015  019F0     .WORD   21, 21          ;
0015  0015  019F4     .WORD   21, 21          ;
0015  0015  019F8     .WORD   21, 21          ;
000E  0000  019FC     .WORD   0, 14           ;
0015  0015  01A00     .WORD   21, 21          ;
0015  0015  01A04     .WORD   21, 21          ;
0015  0015  01A08     .WORD   21, 21          ;
0015  0015  01A0C     .WORD   21, 21          ;
0015  0015  01A10     .WORD   21, 21          ;
0015  0015  01A14     .WORD   21, 21          ;
0015  0015  01A18     .WORD   21, 21          ;
0015  0015  01A1C     .WORD   21, 21          ;
0015  0015  01A20     .WORD   21, 21          ;
0015  0015  01A24     .WORD   21, 21          ;
0015  0015  01A28     .WORD   21, 21          ;
0015  0015  01A2C     .WORD   21, 21          ;
```

```
0015  0015  01A30    .WORD   21, 21          ;
0015  0015  01A34    .WORD   21, 21          ;
0015  0015  01A38    .WORD   21, 21          ;
0015  0015  01A3C    .WORD   21, 21          ;
0015  0015  01A40    .WORD   21, 21          ;
0015  0015  01A44    .WORD   21, 21          ;
0015  0015  01A48    .WORD   21, 21          ;
0015  0015  01A4C    .WORD   21, 21          ;
0015  0015  01A50    .WORD   21, 21          ;
0012  0012  01A54    .WORD   18, 18          ;
0015  0015  01A58    .WORD   21, 21          ;
0015  0015  01A5C    .WORD   21, 21          ;
0015  0015  01A60    .WORD   21, 21          ;
0012  0000  01A64    .WORD   0, 18           ;
0015  0015  01A68    .WORD   21, 21          ;
0015  0015  01A6C    .WORD   21, 21          ;
0015  0015  01A70    .WORD   21, 21          ;
0015  0015  01A74    .WORD   21, 21          ;
0015  0015  01A78    .WORD   21, 21          ;
0015  0015  01A7C    .WORD   21, 21          ;
0015  0015  01A80    .WORD   21, 21          ;
0015  0015  01A84    .WORD   21, 21          ;
0015  0015  01A88    .WORD   21, 21          ;
0015  0015  01A8C    .WORD   21, 21          ;
0015  0015  01A90    .WORD   21, 21          ;
0015  0015  01A94    .WORD   21, 21          ;
0015  0015  01A98    .WORD   21, 21          ;
0015  0015  01A9C    .WORD   21, 21          ;
0015  0015  01AA0    .WORD   21, 21          ;
0015  0015  01AA4    .WORD   21, 21          ;
0015  0015  01AA8    .WORD   21, 21          ;
0013  0013  01AAC    .WORD   19, 19          ;
0015  0015  01AB0    .WORD   21, 21          ;
0015  0015  01AB4    .WORD   21, 21          ;
0015  0015  01AB8    .WORD   21, 21          ;
0015  0015  01ABC    .WORD   21, 21          ;
0013  0000  01AC0    .WORD   0, 19           ;
0014  0000  01AC4    .WORD   0, 20           ;
0015  0015  01AC8    .WORD   21, 21          ;
0015  0015  01ACC    .WORD   21, 21          ;
0015  0015  01AD0    .WORD   21, 21          ;
0015  0015  01AD4    .WORD   21, 21          ;
0015  0015  01AD8    .WORD   21, 21          ;
0015  0015  01ADC    .WORD   21, 21          ;
0015  0015  01AE0    .WORD   21, 21          ;
0015  0015  01AE4    .WORD   21, 21          ;
0015  0015  01AE8    .WORD   21, 21          ;
0015  0015  01AEC    .WORD   21, 21          ;
0015  0015  01AF0    .WORD   21, 21          ;
0015  0015  01AF4    .WORD   21, 21          ;
0015  0015  01AF8    .WORD   21, 21          ;
0015  0015  01AFC    .WORD   21, 21          ;
0015  0015  01B00    .WORD   21, 21          ;
0015  0015  01B04    .WORD   21, 21          ;
0015  0015  01B08    .WORD   21, 21          ;
```

```
      0015  0015  01B0C     .WORD   21, 21          ;
      0015  0015  01B10     .WORD   21, 21          ;
      0015  0015  01B14     .WORD   21, 21          ;
      0015  0015  01B18     .WORD   21, 21          ;
      0015  0015  01B1C     .WORD   21, 21          ;
      0015  0015  01B20     .WORD   21, 21          ;
      0015  0015  01B24     .WORD   21, 21          ;
      0015  0015  01B28     .WORD   21, 21          ;
      0015  0015  01B2C     .WORD   21, 21          ;
      0015  0015  01B30     .WORD   21, 21          ;
      0015  0015  01B34     .WORD   21, 21          ;
      0015  0015  01B38     .WORD   21, 21          ;
      0015  0015  01B3C     .WORD   21, 21          ;
      0015  0015  01B40     .WORD   21, 21          ;
      0015  0015  01B44     .WORD   21, 21          ;
      0015  0015  01B48     .WORD   21, 21          ;
      0015  0015  01B4C     .WORD   21, 21          ;
      0015  0015  01B50     .WORD   21, 21          ;
      0015  0015  01B54     .WORD   21, 21          ;
      0015  0015  01B58     .WORD   21, 21          ;
      0014  0014  01B5C     .WORD   20, 20          ;
      0015  0015  01B60     .WORD   21, 21          ;
      0015  0015  01B64     .WORD   21, 21          ;
      0015  0015  01B68     .WORD   21, 21          ;
      0015  0015  01B6C     .WORD   21, 21          ;
      0013  0000  01B70     .WORD   0, 19           ;
      0014  0000  01B74     .WORD   0, 20           ;
      0015  0015  01B78     .WORD   21, 21          ;
      0015  0015  01B7C     .WORD   21, 21          ;
      0015  0015  01B80     .WORD   21, 21          ;
      0015  0015  01B84     .WORD   21, 21          ;
      0015  0015  01B88     .WORD   21, 21          ;
      0015  0015  01B8C     .WORD   21, 21          ;
      0015  0015  01B90     .WORD   21, 21          ;
      0015  0015  01B94     .WORD   21, 21          ;
      0015  0015  01B98     .WORD   21, 21          ;
      0015  0015  01B9C     .WORD   21, 21          ;
      0015  0015  01BA0     .WORD   21, 21          ;
      0015  0015  01BA4     .WORD   21, 21          ;
      0015  0015  01BA8     .WORD   21, 21          ;
      0015  0015  01BAC     .WORD   21, 21          ;
      0015  0015  01BB0     .WORD   21, 21          ;
      0013  0013  01BB4     .WORD   19, 19          ;
      0015  0015  01BB8     .WORD   21, 21          ;
      0015  0015  01BBC     .WORD   21, 21          ;
      0015  0015  01BC0     .WORD   21, 21          ;
      0015  0015  01BC4     .WORD   21, 21          ;
      0013  0000  01BC8     .WORD   0, 19           ;
      0014  0000  01BCC     .WORD   0, 20           ;
      0015  0015  01BD0     .WORD   21, 21          ;
      0015  0015  01BD4     .WORD   21, 21          ;
      0015  0015  01BD8     .WORD   21, 21          ;
      0015  0015  01BDC     .WORD   21, 21          ;
      0015  0015  01BE0     .WORD   21, 21          ;
      0015  0015  01BE4     .WORD   21, 21          ;
```

```
0015  0015  01BE8     .WORD   21, 21          ;
0015  0015  01BEC     .WORD   21, 21          ;
0015  0015  01BF0     .WORD   21, 21          ;
0015  0015  01BF4     .WORD   21, 21          ;
0015  0015  01BF8     .WORD   21, 21          ;
0015  0015  01BFC     .WORD   21, 21          ;
0015  0015  01C00     .WORD   21, 21          ;
0015  0015  01C04     .WORD   21, 21          ;
0015  0015  01C08     .WORD   21, 21          ;
0015  0015  01C0C     .WORD   21, 21          ;
0015  0015  01C10     .WORD   21, 21          ;
0015  0015  01C14     .WORD   21, 21          ;
0015  0015  01C18     .WORD   21, 21          ;
0015  0015  01C1C     .WORD   21, 21          ;
0015  0015  01C20     .WORD   21, 21          ;
0015  0015  01C24     .WORD   21, 21          ;
0015  0015  01C28     .WORD   21, 21          ;
0015  0015  01C2C     .WORD   21, 21          ;
0015  0015  01C30     .WORD   21, 21          ;
0015  0015  01C34     .WORD   21, 21          ;
0015  0015  01C38     .WORD   21, 21          ;
0015  0015  01C3C     .WORD   21, 21          ;
0015  0015  01C40     .WORD   21, 21          ;
0015  0015  01C44     .WORD   21, 21          ;
0015  0015  01C48     .WORD   21, 21          ;
0015  0015  01C4C     .WORD   21, 21          ;
0015  0015  01C50     .WORD   21, 21          ;
0015  0015  01C54     .WORD   21, 21          ;
0015  0015  01C58     .WORD   21, 21          ;
0015  0015  01C5C     .WORD   21, 21          ;
0015  0015  01C60     .WORD   21, 21          ;
0013  0013  01C64     .WORD   19, 19          ;
0015  0015  01C68     .WORD   21, 21          ;
0015  0015  01C6C     .WORD   21, 21          ;
0015  0015  01C70     .WORD   21, 21          ;
0015  0015  01C74     .WORD   21, 21          ;
0013  0000  01C78     .WORD   0, 19           ;
0014  0000  01C7C     .WORD   0, 20           ;
0015  0015  01C80     .WORD   21, 21          ;
0015  0015  01C84     .WORD   21, 21          ;
0015  0015  01C88     .WORD   21, 21          ;
0015  0015  01C8C     .WORD   21, 21          ;
0015  0015  01C90     .WORD   21, 21          ;
0015  0015  01C94     .WORD   21, 21          ;
0015  0015  01C98     .WORD   21, 21          ;
0015  0015  01C9C     .WORD   21, 21          ;
0015  0015  01CA0     .WORD   21, 21          ;
0015  0015  01CA4     .WORD   21, 21          ;
0015  0015  01CA8     .WORD   21, 21          ;
0015  0015  01CAC     .WORD   21, 21          ;
0015  0015  01CB0     .WORD   21, 21          ;
0015  0015  01CB4     .WORD   21, 21          ;
0015  0015  01CB8     .WORD   21, 21          ;
0013  0013  01CBC     .WORD   19, 19          ;
0015  0015  01CC0     .WORD   21, 21          ;
```

```
0015  0015  01CC4     .WORD   21, 21          ;
0015  0015  01CC8     .WORD   21, 21          ;
0015  0015  01CCC     .WORD   21, 21          ;
0013  0000  01CD0     .WORD   0, 19           ;
0014  0000  01CD4     .WORD   0, 20           ;
0015  0015  01CD8     .WORD   21, 21          ;
0015  0015  01CDC     .WORD   21, 21          ;
0015  0015  01CE0     .WORD   21, 21          ;
0015  0015  01CE4     .WORD   21, 21          ;
0015  0015  01CE8     .WORD   21, 21          ;
0015  0015  01CEC     .WORD   21, 21          ;
0015  0015  01CF0     .WORD   21, 21          ;
0015  0015  01CF4     .WORD   21, 21          ;
0015  0015  01CF8     .WORD   21, 21          ;
0015  0015  01CFC     .WORD   21, 21          ;
0015  0015  01D00     .WORD   21, 21          ;
0015  0015  01D04     .WORD   21, 21          ;
0015  0015  01D08     .WORD   21, 21          ;
0015  0015  01D0C     .WORD   21, 21          ;
0015  0015  01D10     .WORD   21, 21          ;
0013  0013  01D14     .WORD   19, 19          ;
0015  0015  01D18     .WORD   21, 21          ;
0015  0015  01D1C     .WORD   21, 21          ;
0015  0015  01D20     .WORD   21, 21          ;
0015  0015  01D24     .WORD   21, 21          ;
0013  0000  01D28     .WORD   0, 19           ;
0014  0000  01D2C     .WORD   0, 20           ;
0015  0015  01D30     .WORD   21, 21          ;
0015  0015  01D34     .WORD   21, 21          ;
0015  0015  01D38     .WORD   21, 21          ;
0015  0015  01D3C     .WORD   21, 21          ;
0015  0015  01D40     .WORD   21, 21          ;
0015  0015  01D44     .WORD   21, 21          ;
0015  0015  01D48     .WORD   21, 21          ;
0015  0015  01D4C     .WORD   21, 21          ;
0015  0015  01D50     .WORD   21, 21          ;
0015  0015  01D54     .WORD   21, 21          ;
0015  0015  01D58     .WORD   21, 21          ;
0015  0015  01D5C     .WORD   21, 21          ;
0015  0015  01D60     .WORD   21, 21          ;
0015  0015  01D64     .WORD   21, 21          ;
0015  0015  01D68     .WORD   21, 21          ;
0013  0013  01D6C     .WORD   19, 19          ;
0015  0015  01D70     .WORD   21, 21          ;
0015  0015  01D74     .WORD   21, 21          ;
0015  0015  01D78     .WORD   21, 21          ;
0015  0015  01D7C     .WORD   21, 21          ;
0013  0000  01D80     .WORD   0, 19           ;
0014  0000  01D84     .WORD   0, 20           ;
0015  0015  01D88     .WORD   21, 21          ;
0015  0015  01D8C     .WORD   21, 21          ;
0015  0015  01D90     .WORD   21, 21          ;
0015  0015  01D94     .WORD   21, 21          ;
0015  0015  01D98     .WORD   21, 21          ;
0015  0015  01D9C     .WORD   21, 21          ;
```

```
0015  0015  01DA0     .WORD   21, 21          ;
0015  0015  01DA4     .WORD   21, 21          ;
0015  0015  01DA8     .WORD   21, 21          ;
0015  0015  01DAC     .WORD   21, 21          ;
0015  0015  01DB0     .WORD   21, 21          ;
0015  0015  01DB4     .WORD   21, 21          ;
0015  0015  01DB8     .WORD   21, 21          ;
0015  0015  01DBC     .WORD   21, 21          ;
0015  0015  01DC0     .WORD   21, 21          ;
0013  0013  01DC4     .WORD   19, 19          ;
0015  0015  01DC8     .WORD   21, 21          ;
0015  0015  01DCC     .WORD   21, 21          ;
0015  0015  01DD0     .WORD   21, 21          ;
0015  0015  01DD4     .WORD   21, 21          ;
0013  0000  01DD8     .WORD   0, 19           ;
0014  0000  01DDC     .WORD   0, 20           ;
0015  0015  01DE0     .WORD   21, 21          ;
0015  0015  01DE4     .WORD   21, 21          ;
0015  0015  01DE8     .WORD   21, 21          ;
0015  0015  01DEC     .WORD   21, 21          ;
0015  0015  01DF0     .WORD   21, 21          ;
0015  0015  01DF4     .WORD   21, 21          ;
0015  0015  01DF8     .WORD   21, 21          ;
0015  0015  01DFC     .WORD   21, 21          ;
0015  0015  01E00     .WORD   21, 21          ;
0015  0015  01E04     .WORD   21, 21          ;
0015  0015  01E08     .WORD   21, 21          ;
0015  0015  01E0C     .WORD   21, 21          ;
0015  0015  01E10     .WORD   21, 21          ;
0015  0015  01E14     .WORD   21, 21          ;
0015  0015  01E18     .WORD   21, 21          ;
0013  0013  01E1C     .WORD   19, 19          ;
0015  0015  01E20     .WORD   21, 21          ;
0015  0015  01E24     .WORD   21, 21          ;
0015  0015  01E28     .WORD   21, 21          ;
0015  0015  01E2C     .WORD   21, 21          ;
0013  0000  01E30     .WORD   0, 19           ;
0014  0000  01E34     .WORD   0, 20           ;
0015  0015  01E38     .WORD   21, 21          ;
0015  0015  01E3C     .WORD   21, 21          ;
0015  0015  01E40     .WORD   21, 21          ;
0015  0015  01E44     .WORD   21, 21          ;
0015  0015  01E48     .WORD   21, 21          ;
0015  0015  01E4C     .WORD   21, 21          ;
0015  0015  01E50     .WORD   21, 21          ;
0015  0015  01E54     .WORD   21, 21          ;
0015  0015  01E58     .WORD   21, 21          ;
0015  0015  01E5C     .WORD   21, 21          ;
0015  0015  01E60     .WORD   21, 21          ;
0015  0015  01E64     .WORD   21, 21          ;
0015  0015  01E68     .WORD   21, 21          ;
0015  0015  01E6C     .WORD   21, 21          ;
0015  0015  01E70     .WORD   21, 21          ;
0013  0013  01E74     .WORD   19, 19          ;
0015  0015  01E78     .WORD   21, 21          ;
```

```
      0015  0015  01E7C     .WORD   21, 21          ;
      0015  0015  01E80     .WORD   21, 21          ;
      0015  0015  01E84     .WORD   21, 21          ;
      0013  0000  01E88     .WORD   0, 19           ;
      0014  0000  01E8C     .WORD   0, 20           ;
      0015  0015  01E90     .WORD   21, 21          ;
      0015  0015  01E94     .WORD   21, 21          ;
      0015  0015  01E98     .WORD   21, 21          ;
      0015  0015  01E9C     .WORD   21, 21          ;
      0015  0015  01EA0     .WORD   21, 21          ;
      0015  0015  01EA4     .WORD   21, 21          ;
      0015  0015  01EA8     .WORD   21, 21          ;
      0015  0015  01EAC     .WORD   21, 21          ;
      0015  0015  01EB0     .WORD   21, 21          ;
      0015  0015  01EB4     .WORD   21, 21          ;
      0015  0015  01EB8     .WORD   21, 21          ;
      0015  0015  01EBC     .WORD   21, 21          ;
      0015  0015  01EC0     .WORD   21, 21          ;
      0015  0015  01EC4     .WORD   21, 21          ;
      0015  0015  01EC8     .WORD   21, 21          ;
      0015  0015  01ECC     .WORD   21, 21          ;
      0015  0015  01ED0     .WORD   21, 21          ;
      0015  0015  01ED4     .WORD   21, 21          ;
      0015  0015  01ED8     .WORD   21, 21          ;
      0012  0000  01EDC     .WORD   0, 18           ;
      0013  0000  01EE0     .WORD   0, 19           ;
      0014  0000  01EE4     .WORD   0, 20           ;
      0015  0015  01EE8     .WORD   21, 21          ;

          $MODULE=        P.AAA
          CRD$GAW_MM_STATE_TABLE==
                P.AAB
          CRD$GAW_TQ_STATE_TABLE==
                P.AAC
          CRD$GAW_HS_STATE_TABLE==
                P.AAD


;             PSECT SUMMARY
;
; Name     Bytes          Attributes
;
;  DATA          7916  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK            33  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------

$CRD_LITERALS   Macro    Lib03    82
$EQULST    Macro    Lib04    82
$ER    Comptime    98
$HS_ENTRY$   Unbound          291     294     296     298     301     318     320     740     761     769
               782     787     795     803     811     819     827     835     841     849
               857     865     871     879     885     893     901     909     923     937
               943     950     958     961     969     977     982     990     998    1006
              1014    1022    1030    1038    1046
    Comptime   283    740     761     769     782     787     795     803     811     819     827
               835     841     849     857     865     871     879     885     893     901
               909     923     937     943     950     958     961     969     977     982
               990     998    1006    1014    1022    1030    1038    1046
$HS_PREV_COLUMN$  Unbound          335     338     347     353     740     761     769     782     795     803
               811     819     827     835     841     849     857     865     871     879
               885     893     901     909     923     937     943     950     958     969
               977     990     998    1006    1014    1022    1030    1038    1046
    Comptime   281    740     761     769     782     787     795     803     811     819     827
               835     841     849     857     865     871     879     885     893     901
               909     923     937     943     950     958     969     977     982     990
               998    1006    1014    1022    1030    1038    1046
$HS_ROWS$   Unbound          293     299     740     761     769     782     787     795     803     811
               819     827     835     841     849     857     865     871     879     885
               893     901     909     923     937     943     950     958     961     969
               977     982     990     998    1006    1014    1022    1030    1038    1046
    Comptime   284    740     761     769     782     787     795     803     811     819     827
               835     841     849     857     865     871     879     885     893     901
               909     923     937     943     950     958     961     969     977     982
               990     998    1006    1014    1022    1030    1038    1046
$HS_WINDOW_SIZE$  Unbound          338     340     342     353     355     357     740     761     769     782
               787     795     803     811     819     827     835     841     849     857
               865     871     879     885     893     901     909     923     937     943
               950     958     969     977     982     990     998    1006    1014    1022
              1030    1038    1046
    Comptime   282    740     761     769     782     787     795     803     811     819     827
               835     841     849     857     865     871     879     885     893     901
               909     923     937     943     950     958     969     977     982     990
               998    1006    1014    1022    1030    1038    1046
$MM_ENTRY$   Unbound          125     128     130     132     135     152     154     393     411     417
               425     430     435     440     445     450     455     460     465     470
               475     480     485     490     495     500     505     510     518     526
               531     536     541     544     549     554     559     562     565     570
               573     578     583     588     591
    Comptime   117    393     411     417     425     430     435     440     445     450     455
               460     465     470     475     480     485     490     495     500     505
               510     518     526     531     536     541     544     549     554     559
               562     565     570     573     578     583     588     591
$MM_PREV_COLUMN$  Unbound          169     172     181     187     393     411     417     425     518     526
    Comptime   115    393     411     417     425     430     435     440     445     450     455
               460     465     470     475     480     485     490     495     500     505
               510     518     526     531     536     541     549     554     559     570
               578     583     588
$MM_ROWS$   Unbound          127     133     393     411     417     425     430     435     440     445
               450     455     460     465     470     475     480     485     490     495
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
                 500     505     510     518     526     531     536     541     544     549
                 554     559     562     565     570     573     578     583     588     591
    Comptime     118   393     411     417     425     430     435     440     445     450     455
                 460     465     470     475     480     485     490     495     500     505
                 510     518     526     531     536     541     544     549     554     559
                 562     565     570     573     578     583     588     591
$MM_WINDOW_SIZE$  Unbound          172     174     176     187     189     191     393     411     417     425
                 518     526
    Comptime     116   393     411     417     425     430     435     440     445     450     455
                 460     465     470     475     480     485     490     495     500     505
                 510     518     526     531     536     541     549     554     559     570
                 578     583     588
$MODULE    Bind           98
$TQ_ENTRY$   Unbound          208     211     213     215     218     235     237     600     612     615
                 618     621     624     627     630     633     636     639     642     645
                 648     651     654     657     660     663     666     669     672     675
                 678     681     684     687     690     693     696     699     702     705
                 708     711     714     717     720
    Comptime     200   600     612     615     618     621     624     627     630     633     636
                 639     642     645     648     651     654     657     660     663     666
                 669     672     675     678     681     684     687     690     693     696
                 699     702     705     708     711     714     717     720
$TQ_PREV_COLUMN$  Unbound          252     255     264     270     612
    Comptime     198   612
$TQ_ROWS$    Unbound          210     216     600     612     615     618     621     624     627     630
                 633     636     639     642     645     648     651     654     657     660
                 663     666     669     672     675     678     681     684     687     690
                 693     696     699     702     705     708     711     714     717     720
    Comptime     201   600     612     615     618     621     624     627     630     633     636
                 639     642     645     648     651     654     657     660     663     666
                 669     672     675     678     681     684     687     690     693     696
                 699     702     705     708     711     714     717     720
$TQ_WINDOW_SIZE$  Unbound          255     257     259     270     272     274     612
    Comptime     199   612
ACTION     MacrForm     166   179
    MacrForm     249   262
    MacrForm     332   345
ACTION_ROUTINE    MacrForm     121   123
    MacrForm     141   142
    MacrForm     204   206
    MacrForm     224   225
    MacrForm     287   289
    MacrForm     307   308
AUTO$K_EXIT_AUTOSIZER_ERROR Literal      82
AUTO$K_EXIT_CONTROL_C  Literal      82
AUTO$K_EXIT_DUMMY_2  Literal      82
AUTO$K_EXIT_DUMMY_3  Literal      82
AUTO$K_EXIT_ERRORS_COMPLETE Literal      82
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal      82
AUTO$K_EXIT_INTERNAL_ERROR Literal      82
AUTO$K_EXIT_INV_BASE_SYSTEM Literal      82
AUTO$K_EXIT_LAST_REASON  Literal      82     82
AUTO$K_EXIT_NOERRORS_COMPLETE Literal      82
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     82
AUTO$K_EXIT_NOERRORS_PREP Literal     82
COLUMN    MacrForm   166   172     181
    MacrForm   249   255     264
    MacrForm   332   338     347
CRD$GAW_HS_STATE_TABLE  External Lib03
    GlobBind   727
CRD$GAW_MM_STATE_TABLE  External Lib03
    GlobBind   386
CRD$GAW_TQ_STATE_TABLE  External Lib03
    GlobBind   598
CRD$GA_HS_CURRENT_DDB_PTR Global     372 Lib03e
CRD$GA_HS_CURRENT_HSB_PTR Global     373 Lib03e
CRD$GB_CURRENT_STATE_TABLE Global     376 Lib03e
CRD$GL_HS_CURRENT_HSB_NUMB Global     374 Lib03e
CRD$GL_HS_CURRENT_STATE  Global     368 Lib03e
CRD$GL_MM_CURRENT_STATE  Global     366 Lib03e
CRD$GL_PREVIOUS_STATE  Global     364 Lib03e
CRD$GL_TQ_CURRENT_STATE  Global     367 Lib03e
CRD$GL_TQ_CURRENT_TQ_ENTRY Global     370 Lib03e
CRD$K_ACTION_EQL_DO_NOTHING Literal     82
CRD$K_ACTION_RANGE_ERROR Literal     82
CRD$K_ADVANCE_DIAGNOSTIC Literal     82
CRD$K_ADVANCE_GENERAL_COM Literal     82
CRD$K_ADVANCE_STATE  Literal     82
CRD$K_ALLOCATION_ERROR  Literal     82
CRD$K_ASSIGN_PTABLE_PTRS Literal     82
CRD$K_AUTOSIZER_ERROR  Literal     82
CRD$K_AUTOSIZER_SET_FLAGS Literal     82
CRD$K_BAD_ACTION_NUMBER  Literal     82
CRD$K_BAD_TYPECODE_ERROR Literal     82
CRD$K_BASE_SYSTEM_IDC  Literal     82
CRD$K_BASE_SYSTEM_LESI  Literal     82
CRD$K_BASE_SYSTEM_NULL  Literal     82
CRD$K_BASE_SYSTEM_UDA_0  Literal     82
CRD$K_BASE_SYSTEM_UDA_1  Literal     82
CRD$K_BASE_SYSTEM_UDA_2  Literal     82
CRD$K_BASE_SYSTEM_UDA_3  Literal     82
CRD$K_CRD_INITIALIZATION Literal     82
CRD$K_CREATE_HST  Literal     82
CRD$K_DATA_LENGTH_ERROR  Literal     82
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     82
CRD$K_DDB_BLINK   Literal     82
CRD$K_DDB_FLINK   Literal     82
CRD$K_DDB_LAST_ENTRY  Literal     82
CRD$K_DDB_MEMORY_SIZE  Literal     82
CRD$K_DDB_MENU_DEVICE_NUMB Literal     82
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     82
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     82
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     82
CRD$K_DDB_PTABLE_ENTRY  Literal     82
CRD$K_DDB_STATUS  Literal     82
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     82
```

Symbol	Type	Defined	Referenced ...
CRD$K_DEVICE_NAME	Literal	82	
CRD$K_DIAGNOSTIC_ERROR	Literal	82	
CRD$K_DIAGNOSTIC_NAME	Literal	82	
CRD$K_DONE_GENERAL_COMS	Literal	82	
CRD$K_DO_NOTHING	Literal	82	
CRD$K_DUMMY_10	Literal	82	
CRD$K_DUMMY_11	Literal	82	
CRD$K_DUMMY_12	Literal	82	
CRD$K_DUMMY_13	Literal	82	
CRD$K_DUMMY_3	Literal	82	
CRD$K_DUMMY_4	Literal	82	
CRD$K_DUMMY_5	Literal	82	
CRD$K_DUMMY_6	Literal	82	
CRD$K_DUMMY_7	Literal	82	
CRD$K_DUMMY_8	Literal	82	
CRD$K_DUMMY_9	Literal	82	
CRD$K_EXIT_CRD	Literal	82	
CRD$K_HOOK_POINT	Literal	82	
CRD$K_HS_INTERNAL_ERROR	Literal	82	
CRD$K_IDC_EVDLB	Literal	82	
CRD$K_IDC_EVDLC	Literal	82	
CRD$K_IDC_EVDLD	Literal	82	
CRD$K_IDC_EVRAD_0	Literal	82	
CRD$K_IDC_EVRAD_1	Literal	82	
CRD$K_IDC_LAST_DIAGNOSTIC	Literal	82	
CRD$K_INTERNAL_ERROR	Literal	82	
CRD$K_LAST_ACTION_ROUTINE	Literal	82	
CRD$K_LAST_ENTRY	Literal	82	
CRD$K_LAST_FUNCTION	Literal	82	
CRD$K_LAST_HOOK_POINT	Literal	82	
CRD$K_LAST_INTERNAL_ERROR	Literal	82	
CRD$K_LAST_TYPE	Literal	82	
CRD$K_LESI_ENWCA	Literal	82	
CRD$K_LESI_EVRMB_0	Literal	82	
CRD$K_LESI_EVRMB_1	Literal	82	
CRD$K_LESI_LAST_DIAGNOSTIC	Literal	82	
CRD$K_LEVEL_4_PASSED	Literal	82	
CRD$K_LOAD_AUTOSIZER	Literal	82	
CRD$K_LOAD_ERROR	Literal	82	
CRD$K_LOAD_GENERAL_COM	Literal	82	
CRD$K_LOAD_LOAD_COM	Literal	82	
CRD$K_LOAD_SELECT_COM	Literal	82	
CRD$K_LOAD_SET_EVENT_COM	Literal	82	
CRD$K_LOAD_SET_FLAGS_COM	Literal	82	
CRD$K_LOAD_START_COM	Literal	82	
CRD$K_LOAD_SUP_FILE	Literal	82	
CRD$K_MM_INTERNAL_ERROR	Literal	82	
CRD$K_NEW_LINE	Literal	82	
CRD$K_NUMB_TYPECODES	Literal	Lib03	383 594 723 1049
CRD$K_PREPARATION_ERROR	Literal	82	
CRD$K_PREPARATION_ERROR_TEXT	Literal	82	
CRD$K_RUNTIME	Literal	82	
CRD$K_SAME_LINE	Literal	82	

Symbol	Type	Defined	Referenced ...
CRD$K_SET_EVENT_ARGS	Literal	82	
CRD$K_SET_FLAGS_ARGS	Literal	82	
CRD$K_SET_UP_DIAGNOSTIC	Literal	82	
CRD$K_START	Literal	82	
CRD$K_START_AUTOSIZER	Literal	82	
CRD$K_START_ERROR	Literal	82	
CRD$K_START_QUALIFIERS	Literal	82	
CRD$K_STAR_LINE	Literal	82	
CRD$K_STATE_ADVANCE_DIAGNOSTIC	Literal	82	
CRD$K_STATE_ADVANCE_GENERAL_COM	Literal	82	
CRD$K_STATE_ASSIGN_PTABLE_PTRS	Literal	82	
CRD$K_STATE_AUTOSIZER_ERROR	Literal	82	
CRD$K_STATE_AUTOSIZER_RUNNING	Literal	82	
CRD$K_STATE_AUTOSIZER_SET_FLAGS	Literal	82	
CRD$K_STATE_DIAGNOSTIC_ERROR	Literal	82	
CRD$K_STATE_DIAGNOSTIC_RUNNING	Literal	82	
CRD$K_STATE_DONE_DIAGNOSTICS	Literal	82	
CRD$K_STATE_DONE_GENERAL_COMS	Literal	82	
CRD$K_STATE_DUMMY_1	Literal	82	
CRD$K_STATE_DUMMY_10	Literal	82	
CRD$K_STATE_DUMMY_12	Literal	82	
CRD$K_STATE_DUMMY_13	Literal	82	
CRD$K_STATE_DUMMY_2	Literal	82	
CRD$K_STATE_DUMMY_3	Literal	82	
CRD$K_STATE_DUMMY_4	Literal	82	
CRD$K_STATE_DUMMY_6	Literal	82	
CRD$K_STATE_DUMMY_7	Literal	82	
CRD$K_STATE_DUMMY_8	Literal	82	
CRD$K_STATE_EXIT_CRD	Literal	82	
CRD$K_STATE_INTERNAL_ERROR	Literal	82	
CRD$K_STATE_LAST_STATE	Literal	82	
CRD$K_STATE_LEVEL_4_PASSED	Literal	82	
CRD$K_STATE_LOAD_AUTOSIZER	Literal	82	
CRD$K_STATE_LOAD_ERROR	Literal	82	
CRD$K_STATE_LOAD_GENERAL_COM	Literal	82	
CRD$K_STATE_LOAD_LOAD_COM	Literal	82	
CRD$K_STATE_LOAD_SELECT_COM	Literal	82	
CRD$K_STATE_LOAD_SET_EVENT_COM	Literal	82	
CRD$K_STATE_LOAD_SET_FLAGS_COM	Literal	82	
CRD$K_STATE_LOAD_START_COM	Literal	82	
CRD$K_STATE_PREPARATION_ERROR	Literal	82	
CRD$K_STATE_SET_UP_DIAGNOSTIC	Literal	82	
CRD$K_STATE_START	Literal	82	
CRD$K_STATE_START_AUTOSIZER	Literal	82	
CRD$K_STATE_START_ERROR	Literal	82	
CRD$K_TEST_START_TEXT	Literal	82	
CRD$K_TQ_INTERNAL_ERROR	Literal	82	
CRD$K_TYPECODE	Literal	82	
CRD$K_UDA_0_EVDLB	Literal	82	
CRD$K_UDA_0_EVDLC	Literal	82	
CRD$K_UDA_0_EVDLD	Literal	82	
CRD$K_UDA_0_EVRLA_0	Literal	82	
CRD$K_UDA_0_LAST_DIAGNOSTIC	Literal	82	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
CRD$K_UDA_1_EVDLB  Literal     82
CRD$K_UDA_1_EVDLC  Literal     82
CRD$K_UDA_1_EVDLD  Literal     82
CRD$K_UDA_1_EVRLA_0  Literal     82
CRD$K_UDA_1_EVRLA_1  Literal     82
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     82
CRD$K_UDA_2_EVDLB  Literal     82
CRD$K_UDA_2_EVDLC  Literal     82
CRD$K_UDA_2_EVDLD  Literal     82
CRD$K_UDA_2_EVRLA_0  Literal     82
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     82
CRD$K_UDA_3_EVDLB  Literal     82
CRD$K_UDA_3_EVDLC  Literal     82
CRD$K_UDA_3_EVDLD  Literal     82
CRD$K_UDA_3_EVRLA_0  Literal     82
CRD$K_UDA_3_EVRLA_1  Literal     82
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     82
CRD_ASSERT   Macro    Lib03   382
GET1ST_    Macro    Lib04    82
GET2ND_    Unbound          82
HS$FILL_FULL_ROW  Macro      317   961
HS$FILL_N_ENTRYS  Unbound          308     322     342     357     740     761     769     782     787     795
                 803     811     819     827     835     841     849     857     865     871
                 879     885     893     901     909     923     937     943     950     958
                 969     977     982     990     998    1006    1014    1022    1030    1038
                1046
    Macro       287   740     761     769     782     787     795     803     811     819     827
                835     841     849     857     865     871     879     885     893     901
                909     923     937     943     950     958     961     969     977     982
                990     998    1006    1014    1022    1030    1038    1046
HS$INSERT_ENTRYS  Unbound          312     345     740     761     769     782     787     795     803     811
                819     827     835     841     849     857     865     871     879     885
                893     901     909     923     937     943     950     958     969     977
                982     990     998    1006    1014    1022    1030    1038    1046
    Macro       307   740     761     769     782     787     795     803     811     819     827
                835     841     849     857     865     871     879     885     893     901
                909     923     937     943     950     958     969     977     982     990
                998    1006    1014    1022    1030    1038    1046
HS$INSERT_N    Unbound          351     740     761     769     782     787     795     803     811     819
                827     835     841     849     857     865     871     879     885     893
                901     909     923     937     943     950     958     969     977     982
                990     998    1006    1014    1022    1030    1038    1046
    Macro       332   729     740     743     761     764     769     772     782     785     790
                795     798     803     806     811     814     819     822     827     830
                835     838     841     844     849     852     857     860     865     868
                871     874     879     882     885     888     893     896     901     904
                909     912     923     926     937     940     943     946     950     953
                958     964     969     972     977     980     985     990     993     998
               1001    1006    1009    1014    1017    1022    1025    1030    1033    1038
               1041    1046
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal     82   761
HS$K_ACTION_ADVANCE_GENERAL_COM Literal     82   761
HS$K_ACTION_ADVANCE_STATE Literal     82   761
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
HS$K_ACTION_CHANGE_TABLE Literal      82   761
HS$K_ACTION_DIAGNOSTIC_ERROR Literal      82   769      795      803      811      819      827      835      841      849      857
              893      901      909      958      977      990      998     1006     1014     1022
             1030     1038
HS$K_ACTION_DONE_GENERAL_COMS Literal      82   761
HS$K_ACTION_DO_NOTHING  Literal     82   740      769      782      795      803      811      819      827      835      849
              857      865      871      879      885      893      901      909      923      937
              943      950      958      969      977      990      998     1006     1014     1022
             1030     1038     1046
HS$K_ACTION_DUMMY_3  Literal      82
HS$K_ACTION_DUMMY_4  Literal      82
HS$K_ACTION_DUMMY_5  Literal      82
HS$K_ACTION_DUMMY_6  Literal      82
HS$K_ACTION_INIT_HS  Literal      82   761
HS$K_ACTION_INTERNAL_ERROR Literal      82   740      761      769      782      787      795      803      811      819      827
              835      841      849      857      865      871      879      885      893      901
              909      923      937      943      950      958      961      969      977      982
              990      998     1006     1014     1022     1030     1038     1046
HS$K_ACTION_LAST_ACTION  Literal      82
HS$K_ACTION_LOAD_ERROR  Literal     82   782      923
HS$K_ACTION_LOAD_GENERAL_COM Literal      82   761
HS$K_ACTION_LOAD_LOAD_COM Literal     82   761
HS$K_ACTION_LOAD_SELECT_COM Literal      82   761
HS$K_ACTION_LOAD_SET_EVENT_COM Literal      82   761
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal      82   761
HS$K_ACTION_LOAD_START_COM Literal     82   761
HS$K_ACTION_PREPARATION_ERROR Literal     82   969
HS$K_ACTION_START_ERROR  Literal     82   782      923      950
HS$K_NUMB_STATES  Unbound          291      296      322      353      740      761      769      782      787      795
              803      811      819      827      835      841      849      857      865      871
              879      885      893      901      909      923      937      943      950      958
              961      969      977      982      990      998     1006     1014     1022     1030
             1038     1046
    Literal  Lib03   740      761      769      782      787      795      803      811      819      827
              835      841      849      857      865      871      879      885      893      901
              909      923      937      943      950      958      961      969      977      982
              990      998     1006     1014     1022     1030     1038     1046     1049
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     82   761
HS$K_STATE_ADVANCE_GENERAL_COM Literal      82   761
HS$K_STATE_CHANGE_TABLE  Literal     82   761
HS$K_STATE_DIAGNOSTIC_ERROR Literal      82   740      761      769      782      795      803      811      819      827      835
              841      849      857      865      879      893      901      909      923      937
              958      969      977      990      998     1006     1014     1022     1030     1038
             1046
HS$K_STATE DIAGNOSTIC_RUNNING Literal      82   740      761      769      782      795      803      811      819      827      835
              841      849      857      865      871      879      885      893      901      909
              923      937      943      950      958      969      977      990      998     1006
             1014     1022     1030     1038
HS$K_STATE_DONE_GENERAL_COMS Literal      82   761
HS$K_STATE_DUMMY_1  Literal      82
HS$K_STATE_DUMMY_2  Literal      82
HS$K_STATE_DUMMY_3  Literal      82
HS$K_STATE_DUMMY_4  Literal      82
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_STATE_DUMMY_6  Literal      82
HS$K_STATE_INIT_HS  Literal      82   761
HS$K_STATE_INTERNAL_ERROR Literal      82   740     761     769     782     787     795     803     811     819     827
              835     841     849     857     865     871     879     885     893     901
              909     923     937     943     950     958     961     969     977     982
              990     998    1006    1014    1022    1030    1038    1046
HS$K_STATE_LAST_STATE  Literal      82
HS$K_STATE_LOAD_ERROR  Literal      82   740     761     782     923     937
HS$K_STATE_LOAD_GENERAL_COM Literal      82   761
HS$K_STATE_LOAD_LOAD_COM Literal      82   761
HS$K_STATE_LOAD_SELECT_COM Literal      82   761     782     923     937
HS$K_STATE_LOAD_SET_EVENT_COM Literal      82   761     782     923     937
HS$K_STATE_LOAD_SET_FLAGS_COM Literal      82   761     782     923     937
HS$K_STATE_LOAD_START_COM Literal      82   761     782     923     937
HS$K_STATE_PREPARATION_ERROR Literal      82   740     761     769     782     795     803     811     819     827     835
              849     857     865     879     893     901     909     923     937     958
              969     977     990     998    1006    1014    1022    1030    1038    1046
HS$K_STATE_START_ERROR  Literal      82   740     761     782     923     937     950    1046
HS_STATE_TABLE_STRUCTURE Structur Lib03  1049
INTERNAL_ERROR   Unbound           156     176     191     239     259     274     322     342     357     393
              411     417     425     518     526     612     740     761     769     782
              787     795     803     811     819     827     835     841     849     857
              865     871     879     885     893     901     909     923     937     943
              950     958     969     977     982     990     998    1006    1014    1022
             1030    1038    1046
MEN$K_HS_STATE_TABLE  Literal      82
MEN$K_MM_STATE_TABLE  Literal      82
MEN$K_TQ_STATE_TABLE  Literal      82
MENU$K_EXIT_AUTOSIZER_ERROR Literal      82
MENU$K_EXIT_INTERNAL_ERROR Literal      82
MENU$K_EXIT_LAST_REASON  Literal      82
MENU$K_EXIT_OPERATOR_ABORT Literal      82
MENU$K_EXIT_OPERATOR_STOP Literal      82
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal      82
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal      82
MM$FILL_FULL_ROW  Macro      151   544     562     565     573     591
MM$FILL_N_ENTRYS  Unbound           142     156     176     191     393     411     417     425     518     526
    Macro       121   393     411     417     425     430     435     440     445     450     455
              460     465     470     475     480     485     490     495     500     505
              510     518     526     531     536     541     544     549     554     559
              562     565     570     573     578     583     588     591
MM$INSERT_ENTRYS  Unbound           146     179     393     411     417     425     430     435     440     445
              450     455     460     465     470     475     480     485     490     495
              500     505     510     518     526     531     536     541     549     554
              559     570     578     583     588
    Macro       141   393     411     417     425     430     435     440     445     450     455
              460     465     470     475     480     485     490     495     500     505
              510     518     526     531     536     541     549     554     559     570
              578     583     588
MM$INSERT_N    Unbound             185     393     411     417     425     430     435     440     445     450
              455     460     465     470     475     480     485     490     495     500
              505     510     518     526     531     536     541     549     554     559
              570     578     583     588
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
    Macro      166   388     393     396     411     414     417     420     425     428     433
              438    443     448     453     458     463     468     473     478     483
              488    493     498     503     508     513     518     521     526     529
              534    539     547     552     557     568     576     581     586
MM$K_ACTION_ADVANCE_STATE Literal     82   411
MM$K_ACTION_AUTOSIZER_ERROR Literal     82   393     417     425     430     435     440     445     450     455     460
              465    470     475     500     505     510     518     526     536     541
              549    554     578     583     588
MM$K_ACTION_BUILD_DDBS  Literal     82   411
MM$K_ACTION_BUILD_HSTS  Literal     82   411
MM$K_ACTION_CHANGE_TABLE Literal     82   411
MM$K_ACTION_DONE_INITS  Literal     82   411
MM$K_ACTION_DO_NOTHING  Literal     82   393     480     485     490     495     531     559
MM$K_ACTION_DUMMY_3  Literal     82
MM$K_ACTION_DUMMY_4  Literal     82
MM$K_ACTION_DUMMY_5  Literal     82
MM$K_ACTION_DUMMY_6  Literal     82
MM$K_ACTION_DUMMY_7  Literal     82
MM$K_ACTION_DUMMY_8  Literal     82
MM$K_ACTION_INTERNAL_ERROR Literal     82   393     411     417     425     430     435     440     445     450     455
              460    465     470     475     480     485     490     495     500     505
              510    518     526     531     536     541     544     549     554     559
              562    565     570     573     578     583     588     591
MM$K_ACTION_LAST_ACTION  Literal     82
MM$K_ACTION_LOAD_AUTOSIZER Literal     82   411
MM$K_ACTION_MENU_PROMPT  Literal     82   417     570
MM$K_ACTION_PRINT_MENU  Literal     82   411
MM$K_ACTION_PRINT_MENU_HEADING Literal     82   411
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     82   411
MM$K_ACTION_START_AUTOSIZER Literal     82   411
MM$K_NUMB_STATES  Unbound           125     130     156     187     393     411     417     425     430     435
              440    445     450     455     460     465     470     475     480     485
              490    495     500     505     510     518     526     531     536     541
              544    549     554     559     562     565     570     573     578     583
              588    591
    Literal  Lib03   393     411     417     425     430     435     440     445     450     455
              460    465     470     475     480     485     490     495     500     505
              510    518     526     531     536     541     544     549     554     559
              562    565     570     573     578     583     588     591     594
MM$K_STATE_AUTOSIZER_ERROR Literal     82   393     417     425     430     435     440     445     450     455     460
              465    470     475     500     505     510     518     526     536     541
              549    554     578     583     588
MM$K_STATE_AUTOSIZER_RUNNING Literal     82   393     411     417     425     430     435     440     445     450     455
              460    465     470     475     480     485     490     495     500     505
              510    518     526     531     536     541     549     554     578     583
              588
MM$K_STATE_BUILD_DDBS  Literal     82   411
MM$K_STATE_BUILD_HSTS  Literal     82   411
MM$K_STATE_CHANGE_TABLE  Literal     82   411     417     570
MM$K_STATE_DONE_INITS  Literal     82   411
MM$K_STATE_DUMMY_1  Literal     82
MM$K_STATE_DUMMY_2  Literal     82
MM$K_STATE_DUMMY_3  Literal     82
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_STATE_DUMMY_5  Literal      82
MM$K_STATE_DUMMY_7  Literal      82
MM$K_STATE_DUMMY_8  Literal      82
MM$K_STATE_INTERNAL_ERROR Literal      82   393     411     417     425     430     435     440     445     450     455
              460     465     470     475     480     485     490     495     500     505
              510     518     526     531     536     541     544     549     554     559
              562     565     570     573     578     583     588     591
MM$K_STATE_LAST_STATE  Literal      82
MM$K_STATE_LOAD_AUTOSIZER Literal      82   393     411     425     518     526
MM$K_STATE_MENU_PROMPT  Literal      82
MM$K_STATE_PRINT_MENU  Literal      82   411
MM$K_STATE_PRINT_MENU_HEADING Literal      82   411     559
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal      82   393     411     425     518     526
MM$K_STATE_START  Literal      82   411     559
MM$K_STATE_START_AUTOSIZER Literal      82   393     411     425     518     526
MM_STATE_TABLE_STRUCTURE Structur Lib03   594
MODNAM    Macro    Lib01    98
N     MacrForm   121   122     125     130     135
    MacrForm   204   205     208     213     218
    MacrForm   287   288     291     296     301
NEW_STATE   MacrForm   121   123
    MacrForm   141   142
    MacrForm   204   206
    MacrForm   224   225
    MacrForm   287   289
    MacrForm   307   308
NUL2ND_    Macro    Lib04    82
STATE    MacrForm   166   179
    MacrForm   249   262
    MacrForm   332   345
TQ$FILL_FULL_ROW  Macro      234   600     615     618     621     624     627     630     633     636     639
              642     645     648     651     654     657     660     663     666     669
              672     675     678     681     684     687     690     693     696     699
              702     705     708     711     714     717     720
TQ$FILL_N_ENTRYS  Unbound          225     239     259     274     612
    Macro      204   600     612     615     618     621     624     627     630     633     636
              639     642     645     648     651     654     657     660     663     666
              669     672     675     678     681     684     687     690     693     696
              699     702     705     708     711     714     717     720
TQ$INSERT_ENTRYS  Unbound          229     262     612
    Macro      224   612
TQ$INSERT_N   Unbound          268     612
    Macro      249   603     612
TQ$K_ACTION_ADVANCE_STATE Literal      82
TQ$K_ACTION_CHANGE_TABLE Literal      82   612
TQ$K_ACTION_DONE_TEST_QUEUE Literal      82   612
TQ$K_ACTION_DO_NOTHING  Literal      82
TQ$K_ACTION_DUMMY_3  Literal      82
TQ$K_ACTION_DUMMY_4  Literal      82
TQ$K_ACTION_DUMMY_5  Literal      82
TQ$K_ACTION_GET_DDB  Literal      82   612
TQ$K_ACTION_INIT_TQ  Literal      82   612
TQ$K_ACTION_INTERNAL_ERROR Literal      82   600     612     615     618     621     624     627     630     633     636
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
                639     642     645     648     651     654     657     660     663     666
                669     672     675     678     681     684     687     690     693     696
                699     702     705     708     711     714     717     720
TQ$K_ACTION_LAST_ACTION  Literal      82
TQ$K_NUMB_STATES  Unbound           208     213     239     270     600     612     615     618     621     624
                627     630     633     636     639     642     645     648     651     654
                657     660     663     666     669     672     675     678     681     684
                687     690     693     696     699     702     705     708     711     714
                717     720
    Literal  Lib03   600     612     615     618     621     624     627     630     633     636
                639     642     645     648     651     654     657     660     663     666
                669     672     675     678     681     684     687     690     693     696
                699     702     705     708     711     714     717     720     723
TQ$K_STATE_CHANGE_TABLE  Literal      82   612
TQ$K_STATE_DONE_TEST_QUEUE Literal      82   612
TQ$K_STATE_DUMMY_1  Literal      82
TQ$K_STATE_DUMMY_2  Literal      82
TQ$K_STATE_DUMMY_3  Literal      82
TQ$K_STATE_DUMMY_4  Literal      82
TQ$K_STATE_DUMMY_5  Literal      82
TQ$K_STATE_GET_DDB  Literal      82   612
TQ$K_STATE_INIT_TQ  Literal      82   612
TQ$K_STATE_INTERNAL_ERROR Literal      82   600     612     615     618     621     624     627     630     633     636
                639     642     645     648     651     654     657     660     663     666
                669     672     675     678     681     684     687     690     693     696
                699     702     705     708     711     714     717     720
TQ$K_STATE_LAST_STATE  Literal      82
TQ_STATE_TABLE_STRUCTURE Structur Lib03   723


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]MENSTATE.B32;1
    2  Module MENSTATE
   70  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   73  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   76  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   79  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
 1051  Eludom MENSTATE
```

```
 KEY TO REFERENCE TYPE FLAGS

      . Fetch
      = Store
      c Routine call
      a Address use
      a Indirect use
      e External, external routine, or external literal declaration
      f Forward or forward routine declaration
      h Condition handler enabling
      m Map declaration
      u Undeclare declaration


;     Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;            671          1          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;            923          0          0          94          00:00.2
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;            486         21          4          62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0        1000          00:04.3



;      COMMAND QUALIFIERS

; BLISS MENSTATE.B32/LIST=MENSTATE.LIS/CROSS_REFERENCE/DEBUG/OBJECT=MENSTATE.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  0 code + 7949 data bytes
; Run Time:     01:06.0
; Elapsed Time:     01:21.3
; Lines/CPU Min:      955
; Lexemes/CPU-Min: 92332
; Memory Used: 279 pages
; Compilation Complete
```

```
;    0001  0
;    0002  0 %TITLE  'CRD MENU - Test Routines'
;    0003  0 MODULE MENTST  (
;    0004  0    IDENT = '01-04'
;    0005  0    ) =
;    0006  0 !
;    0007  0 ! COPYRIGHT (C)  1982
;    0008  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0009  0 !
;    0010  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0011  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0012  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0013  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0014  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0015  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0016  0 ! REMAIN IN DEC.
;    0017  0 !
;    0018  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0019  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0020  0 ! CORPORATION.
;    0021  0 !
;    0022  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0023  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0024  0 !
```

```
;    0025  0 !++
;    0026  0 ! FACILITY:
;    0027  0 !
;    0028  0 ! ABSTRACT:
;    0029  0 !
;    0030  0 ! ENVIRONMENT:
;    0031  0 !
;    0032  0 ! AUTHOR:
;    0033  0 !
;    0034  0 ! John Ciukaj  May-1982
;    0035  0 !
;    0036  0 ! CREATION DATE:
;    0037  0 !
;    0038  0 ! VERSION:
;    0039  0 !
;    0040  0 ! Modified by:
;    0041  0 !
;    0042  0 ! 01 Jack Stansbury, 03-Mar-1983, Version 1.2
;    0043  0 !  Made the MEN$Preparation_Err_Text routine more robust.
;    0044  0 !
;    0045  0 ! 02 John Ciukaj 27-Apr-1983 version 1.2
;    0046  0 !  Made the MEN$Preparation_Err_Text routine LESS robust.
;    0047  0 !
;    0048  0 ! 03 Scott Cohen 28-Jun-1983 Version 1.2
;    0049  0 !  Added soft console support
;    0050  0 !  - routine MEN$Menu_Summary,
;    0051  0 !
;    0052  0 ! 04 Scott Cohen 23-Sep-1983 Version 1.3
;    0053  0 !  Added new soft console terminal (TT$_VS125) the PEARL
;    0054  0 !
;    0055  0 !--
```

```
;    0056  0 %SBTTL 'Libraries'
;    0057  1 BEGIN
;    0058  1
;    0059  1
;    0060  1 LIBRARY '$DS';
;    0061  1 LIBRARY '$DIAG';
;    0062  1 LIBRARY '$CRD';
;    0063  1 LIBRARY 'SYS$LIBRARY:LIB';
```

```
;    0064  1 %SBTTL 'Forward-External  Routines'
;    0065  1
;    0066  1 FORWARD ROUTINE
;    0067  1  MEN$Scratch_Media_Msg   : ADDRESSING_MODE (LONG_RELATIVE),
;    0068  1  MEN$FncTst_CmplErr_Status  : ADDRESSING_MODE (LONG_RELATIVE),
;    0069  1  MEN$FncTst_TestStrt_Text  : ADDRESSING_MODE (LONG_RELATIVE),
;    0070  1  MEN$Print_Failed_Hdwr   : ADDRESSING_MODE (LONG_RELATIVE),
;    0071  1  MEN$Print_Runtime_Total  : ADDRESSING_MODE (LONG_RELATIVE),
;    0072  1  MEN$CVT_Time_ASC_D   : ADDRESSING_MODE (LONG_RELATIVE),
;    0073  1  MEN$Scan_Numeric_ASC_D   : ADDRESSING_MODE (LONG_RELATIVE),
;    0074  1  MEN$Preparation_Err_Text  : ADDRESSING_MODE (LONG_RELATIVE),
;    0075  1  MEN$Menu_Summary   : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    0076  1 %SBTTL 'Psect Definitions'
;    0077  1
;    0078  1 PSECT
;    0079  1     Plit = Data (NoExecute, Share, NoWrite, Addressing_Mode (Long_Relative));
;    0080  1 PSECT
;    0081  1     Code =  Code (Execute, Share, NoWrite, Addressing_Mode (Long_Relative), PIC);
;    0082  1 PSECT
;    0083  1     Own  = Work (NoExecute, NoShare, Write, Addressing_Mode (Long_Relative));
;    0084  1 PSECT
;    0085  1     Global = Work (NoExecute, NoShare, Write, Addressing_Mode (Long_Relative));
```

```
;    0086  1 %SBTTL 'Module-Global Static Storage'
;    0087  1
;    0088  1      Modnam ('MENTST');    ! Module Name for Errsup Macro
```

```
;    0089  1 %SBTTL 'Own Storage'
;    0090  1
;    0091  1 LITERAL
;    0092  1  DevNam_String_Buf_Sz = 256;
;    0093  1 OWN
;    0094  1  DevNam_String_Buf : VECTOR [DevNam_String_Buf_Sz, BYTE];
```

```
;    0095  1 %SBTTL 'Macro - Literal Definitions'
;    0096  1
;    0097  1  $CRD_Literals;
;    0098  1  $MEN_Literals;
;    0099  1  $SCREEN_Literals;
```

```
;    0100  1 %SBTTL 'MEN$FncTst_CmplErr_Status Routine'
;    0101  1
;    0102  1 GLOBAL ROUTINE MEN$FncTst_CmplErr_Status =
;    0103  1
;    0104  1 !++
;    0105  1 ! FUNCTIONAL DESCRIPTION:
;    0106  1 !
;    0107  1 !    Determines the Completion and Error status of the CRD MENU Functional Test run, and
;    0108  1 !    prints formatted messages.
;    0109  1 !
;    0110  1 !    The test run is considered 'complete' if the DDB Status bit DDB$V_Status_Dev_Test_Complete
;    0111  1 !    is set, or true, for every DDB found in the Test Queue Table (MEN$GA_Test_Queue).
;    0112  1 !
;    0113  1 !    The test run is considered to have a 'no error' status if the DDB$V_Status_Device_Bad
;    0114  1 !    is clear, or false, for every DDB found in the Test Queue Table (MEN$GA_Test_Queue).
;    0115  1 !
;    0116  1 !
;    0117  1 ! INPUT PARAMETERS
;    0118  1 !
;    0119  1 ! OUTPUT PARAMETERS
;    0120  1 !
;    0121  1 ! IMPLICIT INPUTS
;    0122  1 !
;    0123  1 !    Test Queue table pointed to by MEN$GA_Test_Queue.
;    0124  1 !
;    0125  1 ! EXPLICIT OUTPUTS
;    0126  1 !
;    0127  1 !    MEN$GB_Test_In_Progress is specified false at the completion
;    0128  1 !    of this routine.
;    0129  1 !
;    0130  1 ! SIDE EFFECTS
;    0131  1 !
;    0132  1 !
;    0133  1 ! COMPLETION CODES
;    0134  1 !
;    0135  1 !    CRD$K_Success
;    0136  1 !--
```

```
;    0137  2 BEGIN
;    0138  2     REGISTER
;    0139  2  FncTst_Complete  : INITIAL (True), ! Start with 'complete' status
;    0140  2   FncTst_Error   : INITIAL (False), ! Start with 'no error' status
;    0141  2  TstQ_Tbl_Index,
;    0142  2  TstQ_Tbl_Entries,
;    0143  2  TstQ_Tbl_Ptr   : REF VECTOR [, LONG],
;    0144  2  DDB_Ptr    : REF Device_data_Block_Structure,
;    0145  2  DDB_Status;
;    0146  2
;    0147  2     !+
;    0148  2     ! Set up for indexing into Test Queue Table
;    0149  2     !-
;    0150  2
;    0151  2     TstQ_Tbl_Index    = 0;
;    0152  2     TstQ_Tbl_Entries = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb_Entries];
;    0153  2     TstQ_Tbl_Ptr      = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr];
;    0154  2
;    0155  2     !+
;    0156  2     ! Get a DDB pointer from the Test Queue Table until the 'complete'
;    0157  2     ! status and 'error' status is determined
;    0158  2     !-
;    0159  2
;    0160  2     WHILE (.TstQ_Tbl_Index LSS .TstQ_Tbl_Entries) AND ((DDB_Ptr = .TstQ_Tbl_Ptr [.TstQ_Tbl_Index]) NEQ 0) DO
;    0161  3  BEGIN
;    0162  3  DDB_Status       = .DDB_Ptr [CRD$K_DDB_Status];
;    0163  3  IF (.CRD$GB_CRD_Debug) THEN
;    0164  3   $CRD_Print ($ASCIC ('MEN$FncTst_CmplErr_Status: Status = !XL(X)!/'), .DDB_Status);
;    0165  3
;    0166  3  FncTst_Complete = .FncTst_Complete AND .DDB_Status <DDB$V_Status_Dev_Test_Complete>;
;    0167  3  FncTst_Error    = .FncTst_Error    OR  .DDB_Status <DDB$V_Status_Device_Bad>;
;    0168  3  TstQ_Tbl_Index  = .TstQ_Tbl_Index + 1;
;    0169  2  END;
;    0170  2
;    0171  2     !+
;    0172  2     ! Based upon the resultant 'complete' and 'error' status print appropriate messages
;    0173  2     !-
;    0174  2
;    0175  2     SELECTONE .FncTst_Complete OF
;    0176  2  SET
;    0177  2  [True]:
;    0178  2      SELECTONE .FncTst_Error OF
;    0179  2   SET
;    0180  2   [True]:
;    0181  3       BEGIN
;    0182  3       !
;    0183  3       ! Complete with errors
;    0184  3       !
;    0185  3       $CRD_Message (New_Line, MEN$GT_FncTst_Cmpl_Err);
;    0186  3       MEN$Print_Failed_Hdwr ();
;    0187  3       $CRD_Message (New_Line, MEN$GT_Call_FldSrv);
;    0188  2       END;
;    0189  2
;    0190  2   [False]:
;    0191  3       BEGIN
```

```
;    0192  3        !
;    0193  3        ! Complete, No Errors
;    0194  3        !
;    0195  3        $CRD_Message (New_Line, MEN$GT_FncTst_Cmpl_NoErr);
;    0196  2        END;
;    0197  2    TES;
;    0198  2
;    0199  2   [False]:
;    0200  2        SELECTONE .FncTst_Error OF
;    0201  2    SET
;    0202  2    [True]:
;    0203  3        BEGIN
;    0204  3        !
;    0205  3        ! Incomplete, with Errors
;    0206  3        !
;    0207  3        $CRD_Message (New_Line, MEN$GT_FncTst_InCmpl_Err);
;    0208  3        MEN$Print_Failed_Hdwr ();
;    0209  3        $CRD_Message (New_Line, MEN$GT_Call_FldSrv);
;    0210  2        END;
;    0211  2
;    0212  2    [False]:
;    0213  3        BEGIN
;    0214  3        !
;    0215  3        ! Incomplete, No Errors
;    0216  3        !
;    0217  3        $CRD_Message (New_Line, MEN$GT_FncTst_InCmpl_NoErr);
;    0218  3        $CRD_Message (New_Line, MEN$GT_Assistance_Msg);
;    0219  2        END;
;    0220  2    TES;
;    0221  2   TES;
;    0222  2      MEN$GB_Test_In_Progress = False;
;    0223  2      !Indicate that testing is no longer considered in progress
;    0224  2
;    0225  2      !+
;    0226  2      ! Print a menu summary when testing all devices
;    0227  2      ! NOTE: We are testing all devices whenever we
;    0228  2      ! are not testing only 1 device.
;    0229  2      !-
;    0230  3      IF (.TstQ_Tbl_Ptr [1] NEQ 0)
;    0231  2      THEN
;    0232  2   MEN$Menu_Summary ();
;    0233  2
;    0234  2      !+
;    0235  2      ! Soft console - pause and clear the screen before returning to MAIN MENU.
;    0236  2      !-
;    0237  2
;    0238  2      CRD$Screen_Pause (SCREEN$K_Press_Return_MM);        ! [03]
;    0239  2
;    0240  2
;    0241  2      RETURN (CRD$K_Success);
;    0242  1 END;


            .TITLE  MENTST CRD MENU - Test Routines
            .IDENT  \01-04\
```

```
                .PSECT  WORK,NOEXE,2

                00000 DEVNAM_STRING_BUF:
                .BLKB   256

                .PSECT  DATA,NOWRT,NOEXE,  SHR,2

    54  53  54  4E  45  4D  06  00000 P.AAA:   .ASCII  <6>\MENTST\          ;
70  6D  43  5F  74  73  54  63  6E  46  24  4E  45  4D  2C  00007 P.AAB:   .ASCII  \,MEN$FncTst_CmplErr_Status: Status = !XL\  ;
74  53  20  3A  73  75  74  61  74  53  5F  72  72  45  6C  00016           ;
      4C  58  21  20  3D  20  73  75  74  61  00025                 ;
     2F  21  29  58  28  0002F     .ASCII  \(X)!/\             ;

               $MODULE=        P.AAA
                .EXTRN  MEN$SCRATCH_MEDIA_MSG
                .EXTRN  MEN$FNCTST_CMPLERR_STATUS
                .EXTRN  MEN$FNCTST_TESTSTRT_TEXT
                .EXTRN  MEN$PRINT_FAILED_HDWR
                .EXTRN  MEN$PRINT_RUNTIME_TOTAL
                .EXTRN  MEN$CVT_TIME_ASC_D
                .EXTRN  MEN$SCAN_NUMERIC_ASC_D
                .EXTRN  MEN$PREPARATION_ERR_TEXT
                .EXTRN  MEN$MENU_SUMMARY
                .EXTRN  MEN$GA_TEST_QUEUE
                .EXTRN  CRD$GB_CRD_DEBUG
                .EXTRN  CRD$PRINT, MEN$GT_FNCTST_CMPL_ERR
                .EXTRN  CRD$GT_NEW_LINE_MESSAGE
                .EXTRN  MEN$GT_CALL_FLDSRV
                .EXTRN  MEN$GT_FNCTST_CMPL_NOERR
                .EXTRN  MEN$GT_FNCTST_INCMPL_ERR
                .EXTRN  MEN$GT_FNCTST_INCMPL_NOERR
                .EXTRN  MEN$GT_ASSISTANCE_MSG
                .EXTRN  MEN$GB_TEST_IN_PROGRESS
                .EXTRN  CRD$SCREEN_PAUSE

                .PSECT  CODE,NOWRT,  SHR,  PIC,2

                  01FC 00000     .ENTRY  MEN$FNCTST_CMPLERR_STATUS, Save R2,R3,R4,-  ; 0102
                 R5,R6,R7,R8          ;
                52       01  7D 00002     MOVQ    #1, FNCTST_COMPLETE          ; 0137
                54  D4 00005     CLRL    TSTQ_TBL_INDEX          ; 0151
                55 00000000G  EF  7D 00007     MOVQ    MEN$GA_TEST_QUEUE, TSTQ_TBL_ENTRIES        ; 0152
                33  11 0000E     BRB     3$              ; 0160
                58 14   A7  D0 00010 1$:    MOVL    20(DDB_PTR), DDB_STATUS         ; 0162
                13 00000000G  EF  E9 00014     BLBC    CRD$GB_CRD_DEBUG, 2$          ; 0163
                58  DD 0001B     PUSHL   DDB_STATUS              ; 0164
            00000000'  EF  9F 0001D     PUSHAB  P.AAB               ;
                01  DD 00023     PUSHL   #1              ;
                1A  DD 00025     PUSHL   #26             ;
            00000000G  FF       04  FB 00027     CALLS   #4, @CRD$PRINT          ;
        50          58          01      05  EF 0002E 2$:   EXTZV   #5, #1, DDB_STATUS, R0         ; 0166
                50      50  D2 00033     MCOML   R0, R0              ;
                52      50  CA 00036     BICL2   R0, FNCTST_COMPLETE         ;
        50          58          01      04  EF 00039     EXTZV   #4, #1, DDB_STATUS, R0         ; 0167
```

```
                53      50  C8 0003E    BISL2   R0, FNCTST_ERROR         ;
                54  D6 00041    INCL    TSTQ_TBL_INDEX          ; 0168
                55      54  D1 00043 3$:   CMPL    TSTQ_TBL_INDEX, TSTQ_TBL_ENTRIES        ; 0160
                06  18 00046    BGEQ    4$              ;
                57    6644  D0 00048    MOVL    (TSTQ_TBL_PTR)[TSTQ_TBL_INDEX], DDB_PTR     ;
                C2  12 0004C    BNEQ    1$              ;
                01      52  D1 0004E 4$:   CMPL    FNCTST_COMPLETE, #1         ; 0177
                67  12 00051    BNEQ    8$              ;
                01      53  D1 00053    CMPL    FNCTST_ERROR, #1            ; 0180
                42  12 00056    BNEQ    6$              ;
           00000000G  EF  9F 00058    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0185
                01  DD 0005E    PUSHL   #1              ;
                1A  DD 00060    PUSHL   #26             ;
           00000000G  FF       03  FB 00062    CALLS   #3, @CRD$PRINT          ;
           00000000G  EF  9F 00069    PUSHAB  MEN$GT_FNCTST_CMPL_ERR          ;
                01  DD 0006F 5$:   PUSHL   #1              ;
                1A  DD 00071    PUSHL   #26             ;
           00000000G  FF       03  FB 00073    CALLS   #3, @CRD$PRINT          ;
           00000000V  EF       00  FB 0007A    CALLS   #0, MEN$PRINT_FAILED_HDWR       ; 0186
           00000000G  EF  9F 00081    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0187
                01  DD 00087    PUSHL   #1              ;
                1A  DD 00089    PUSHL   #26             ;
           00000000G  FF       03  FB 0008B    CALLS   #3, @CRD$PRINT          ;
           00000000G  EF  9F 00092    PUSHAB  MEN$GT_CALL_FLDSRV          ;
                7F  11 00098    BRB     10$             ;
                53  D5 0009A 6$:   TSTL    FNCTST_ERROR            ; 0190
                03  13 0009C    BEQL    7$              ;
              0083  31 0009E    BRW     11$             ;
           00000000G  EF  9F 000A1 7$:   PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0195
                01  DD 000A7    PUSHL   #1              ;
                1A  DD 000A9    PUSHL   #26             ;
           00000000G  FF       03  FB 000AB    CALLS   #3, @CRD$PRINT          ;
           00000000G  EF  9F 000B2    PUSHAB  MEN$GT_FNCTST_CMPL_NOERR        ;
                5F  11 000B8    BRB     10$             ;
                52  D5 000BA 8$:   TSTL    FNCTST_COMPLETE         ; 0199
                66  12 000BC    BNEQ    11$             ;
                01      53  D1 000BE    CMPL    FNCTST_ERROR, #1            ; 0202
                19  12 000C1    BNEQ    9$              ;
           00000000G  EF  9F 000C3    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0207
                01  DD 000C9    PUSHL   #1              ;
                1A  DD 000CB    PUSHL   #26             ;
           00000000G  FF       03  FB 000CD    CALLS   #3, @CRD$PRINT          ;
           00000000G  EF  9F 000D4    PUSHAB  MEN$GT_FNCTST_INCMPL_ERR        ;
                93  11 000DA    BRB     5$              ;
                53  D5 000DC 9$:   TSTL    FNCTST_ERROR            ; 0212
                44  12 000DE    BNEQ    11$             ;
           00000000G  EF  9F 000E0    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0217
                01  DD 000E6    PUSHL   #1              ;
                1A  DD 000E8    PUSHL   #26             ;
           00000000G  FF       03  FB 000EA    CALLS   #3, @CRD$PRINT          ;
           00000000G  EF  9F 000F1    PUSHAB  MEN$GT_FNCTST_INCMPL_NOERR          ;
                01  DD 000F7    PUSHL   #1              ;
                1A  DD 000F9    PUSHL   #26             ;
           00000000G  FF       03  FB 000FB    CALLS   #3, @CRD$PRINT          ;
           00000000G  EF  9F 00102    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0218
```

```
            01  DD 00108    PUSHL   #1              ;
            1A  DD 0010A    PUSHL   #26             ;
      00000000G  FF      03  FB 0010C    CALLS   #3, @CRD$PRINT          ;
      00000000G  EF  9F 00113    PUSHAB  MEN$GT_ASSISTANCE_MSG           ;
            01  DD 00119 10$:   PUSHL   #1             ;
            1A  DD 0011B    PUSHL   #26             ;
      00000000G  FF      03  FB 0011D    CALLS   #3, @CRD$PRINT          ;
      00000000G  EF  94 00124 11$:   CLRB    MEN$GB_TEST_IN_PROGRESS         ; 0222
     04   A6  D5 0012A    TSTL    4(TSTQ_TBL_PTR)         ; 0230
            07  13 0012D    BEQL    12$             ;
      00000000V  EF      00  FB 0012F    CALLS   #0, MEN$MENU_SUMMARY            ; 0232
            03  DD 00136 12$:   PUSHL   #3              ; 0238
      00000000G  EF      01  FB 00138    CALLS   #1, CRD$SCREEN_PAUSE            ;
            50      01  D0 0013F    MOVL    #1, R0          ; 0241
        04 00142    RET                 ; 0242

; Routine Size:  323 bytes,    Routine Base:  CODE + 0000
```

```
;    0243  1 %SBTTL 'MEN$FncTst_TestStrt_Text Routine'
;    0244  1
;    0245  1 GLOBAL ROUTINE MEN$FncTst_TestStrt_Text (I_DDB, I_HST)  =
;    0246  1
;    0247  1 !++
;    0248  1 ! FUNCTIONAL DESCRIPTION:
;    0249  1 !
;    0250  1 !    Print 'test start text' message for CRD MENU TEST
;    0251  1 !
;    0252  1 ! INPUT PARAMETERS
;    0253  1 !
;    0254  1 !    I_DDB   Address of DDB
;    0255  1 !
;    0256  1 !    I_HST  Address of Hardware Support Table
;    0257  1 !
;    0258  1 ! OUTPUT PARAMETERS
;    0259  1 !
;    0260  1 ! IMPLICIT INPUTS
;    0261  1 !
;    0262  1 ! EXPLICIT OUTPUTS
;    0263  1 !
;    0264  1 !
;    0265  1 ! SIDE EFFECTS
;    0266  1 !
;    0267  1 !
;    0268  1 ! COMPLETION CODES
;    0269  1 !
;    0270  1 !    CRD$K_Success
;    0271  1 !--
```

```
;    0272  2 BEGIN
;    0273  2     LOCAL
;    0274  2  TstCla_Name_Ptr;
;    0275  2
;    0276  2     REGISTER
;    0277  2  Name_Ptr : REF VECTOR [, BYTE],
;    0278  2  DDB_Ptr  : REF Device_Data_Block_Structure,
;    0279  2  HST_Ptr  : REF VECTOR [, LONG],
;    0280  2  SupBlk_Ptr : REF $MEN_SupBlk_ATTR,
;    0281  2  PTable_Ptr : REF $DS_HPO_DECL (),
;    0282  2
;    0283  2  Hdwr_Name_Ptr,
;    0284  2  Dev_Name,
;    0285  2  Runtime_Ptr;
;    0286  2
;    0287  2     BIND
;    0288  2      DDB = .I_DDB,
;    0289  2  HST = .I_HST ;
;    0290  2
;    0291  2     DDB_Ptr = DDB;  ! Point to the DDB
;    0292  2     HST_Ptr = HST;  ! Point to Hardware Support Table
;    0293  2
;    0294  2     !+
;    0295  2     ! Set up for printing Hardware Device Name
;    0296  2     !-
;    0297  2
;    0298  2     PTable_Ptr     = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    0299  2     ! Point to the P-Table
;    0300  2     Hdwr_Name_Ptr = PTable_Ptr [HP$T_TYPE];
;    0301  2     ! Point to the Hardware device name ASCIC
;    0302  2
;    0303  2     !+
;    0304  2     ! Set up for printing Test Class Name.  Use name specified as 'test start text'
;    0305  2     ! in  HST, otherwise use the device test class name specified in support block.
;    0306  2     !-
;    0307  2
;    0308  2     Name_Ptr = .HST_Ptr [CRD$K_Test_Start_Text];
;    0309  2
;    0310  3     IF ((.Name_Ptr EQL 0) OR (.Name_Ptr [0] EQL 0))
;    0311  2     THEN
;    0312  3  BEGIN   ! 0 pointer or Null ASCIC
;    0313  3         SupBLk_Ptr = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr];
;    0314  3     ! Point to the support block
;    0315  3         MEN$Get_TstCla_Name (SupBlk_Ptr [SupBlk$B_Test_Class], TstCla_Name_Ptr);
;    0316  3     ! Get the test class name ASCIC
;    0317  3  END
;    0318  2     ELSE
;    0319  2  TstCla_Name_Ptr = .Name_Ptr;
;    0320  2
;    0321  2     !+
;    0322  2     ! Set up for printing Generic Device Name
;    0323  2     !-
;    0324  2
;    0325  2     Dev_Name = CRD$Get_Device_Name (.PTable_Ptr);
;    0326  2     ! Generic device name ASCIC
```

```
;    0327  2
;    0328  2      !+
;    0329  2      ! Set up for printing Diagnostic Runtime
;    0330  2      !-
;    0331  2
;    0332  2      RunTime_Ptr = .HST_Ptr [CRD$K_RunTime];
;    0333  2      ! Point to Run Time ASCIC
;    0334  2
;    0335  2      !+
;    0336  2      ! Print the test start text
;    0337  2      !-
;    0338  2
; P  0339  2      $CRD_Print (MEN$GT_FncTst_TestStrt_Text,
; P  0340  2        .Hdwr_Name_Ptr, ! Hardware Name ASCIC
; P  0341  2        .TstCla_Name_Ptr, ! Test Class Name ASCIC
; P  0342  2        .Dev_Name,  ! Generic Device name ASCIC
; P  0343  2        .Runtime_Ptr  ! Run time ASCIC
;    0344  2    );
;    0345  2      RETURN (CRD$K_Success);
;    0346  1  END;


            .EXTRN  MEN$GET_TSTCLA_NAME
            .EXTRN  CRD$GET_DEVICE_NAME
            .EXTRN  MEN$GT_FNCTST_TESTSTRT_TEXT

              001C 00000     .ENTRY  MEN$FNCTST_TESTSTRT_TEXT, Save R2,R3,R4      ; 0245
           5E      04  C2 00002     SUBL2   #4, SP                ;
           51 04   AC  7D 00005     MOVQ    I_DDB, DDB_PTR        ; 0291
           53 08   A1  D0 00009     MOVL    8(DDB_PTR), PTABLE_PTR        ; 0298
           54 26   A3  9E 0000D     MOVAB   38(R3), HDWR_NAME_PTR         ; 0300
           50 14   A2  D0 00011     MOVL    20(HST_PTR), NAME_PTR         ; 0308
           04  13 00015     BEQL    1$              ; 0310
           60  95 00017     TSTB    (NAME_PTR)              ;
           12  12 00019     BNEQ    2$              ;
           50 20   A1  D0 0001B 1$:    MOVL    32(DDB_PTR), SUPBLK_PTR       ; 0313
           5E  DD 0001F     PUSHL   SP              ; 0315
       11  A0  9F 00021     PUSHAB  17(SUPBLK_PTR)          ;
        00000000G  EF       02  FB 00024     CALLS   #2, MEN$GET_TSTCLA_NAME       ;
           03  11 0002B     BRB     3$              ; 0310
           6E      50  D0 0002D 2$:    MOVL    NAME_PTR, TSTCLA_NAME_PTR       ; 0319
           53  DD 00030 3$:    PUSHL   PTABLE_PTR              ; 0325
        00000000G  EF       01  FB 00032     CALLS   #1, CRD$GET_DEVICE_NAME       ;
           51 18   A2  D0 00039     MOVL    24(HST_PTR), RUNTIME_PTR      ; 0332
           03  BB 0003D     PUSHR   #^M<R0,R1>              ; 0344
       08  AE  DD 0003F     PUSHL   TSTCLA_NAME_PTR         ;
           54  DD 00042     PUSHL   HDWR_NAME_PTR           ;
        00000000G  EF  9F 00044     PUSHAB  MEN$GT_FNCTST_TESTSTRT_TEXT       ;
           01  DD 0004A     PUSHL   #1              ;
           1A  DD 0004C     PUSHL   #26             ;
        00000000G  FF       07  FB 0004E     CALLS   #7, @CRD$PRINT        ;
           50      01  D0 00055     MOVL    #1, R0          ; 0345
            04 00058     RET             ; 0346
```

```
ZZ-ENSAB-2.0    MEN$FncTst_TestStrt_Text Routine                19-Sep-1985      Fiche 8  Frame C4        Sequence 1483
MENTST  CRD MENU - Test Routines   19-Sep-1985 17:21:39 VAX-11 Bliss-32 V4.1-746     Page  18
01-04  MEN$FncTst_TestStrt_Text Routine   3-Jan-1985 17:02:36 [DS.CRD.V020]MENTST.B32;1  (12)

; Routine Size:  89 bytes,    Routine Base:  CODE + 0143
```

```
;    0347  1 %SBTTL 'MEN$Print_Failed_Hdwr Routine'
;    0348  1
;    0349  1 GLOBAL ROUTINE MEN$Print_Failed_Hdwr =
;    0350  1
;    0351  1 !++
;    0352  1 ! FUNCTIONAL DESCRIPTION:
;    0353  1 !
;    0354  1 !    Print ASCII hardware generic device names of all DDB's in the Test Queue
;    0355  1 !    which indicate that the hardware has failed testing.  A hardware device is
;    0356  1 !    considered failed if the DDB's status bit DDB$V_Status_Device_Bad
;    0357  1 !    is true indicating that CRD MENU has printed a ** FAILED ** when
;    0358  1 !    testing the device.
;    0359  1 !
;    0360  1 !    If no DDB's are indicated as failed then no typeout is received.
;    0361  1 !
;    0362  1 ! INPUT PARAMETERS
;    0363  1 !
;    0364  1 ! OUTPUT PARAMETERS
;    0365  1 !
;    0366  1 ! IMPLICIT INPUTS
;    0367  1 !
;    0368  1 !    Test Queue table pointed to by MEN$GA_Test_Queue.
;    0369  1 !
;    0370  1 ! EXPLICIT OUTPUTS
;    0371  1 !
;    0372  1 !    ASCII text printout if at least one device is found to be bad:
;    0373  1 !
;    0374  1 !    FORMAT:
;    0375  1 !
;    0376  1 !    FAILED HARDWARE: Device name 1, Device name 2, ....., Device name N
;    0377  1 !
;    0378  1 !
;    0379  1 ! SIDE EFFECTS
;    0380  1 !
;    0381  1 !
;    0382  1 ! COMPLETION CODES
;    0383  1 !
;    0384  1 !    CRD$K_Success
;    0385  1 !--
```

```
;    0386  2 BEGIN
;    0387  2     LOCAL
;    0388  2  TstQ_Tbl_Index,
;    0389  2  TstQ_Tbl_Entries,
;    0390  2  TstQ_Tbl_Ptr    : REF VECTOR [, LONG];
;    0391  2
;    0392  2     REGISTER
;    0393  2  Failed_Hdwr_Found : INITIAL (False), ! Assume no failed hardware was found
;    0394  2
;    0395  2  Char_Ptr,
;    0396  2  Numb_Bad_Devices : INITIAL (0),
;    0397  2
;    0398  2  DDB_Ptr    : REF Device_Data_Block_Structure,
;    0399  2  PTable_Ptr    : REF $DS_HPO_DECL (),
;    0400  2  DDB_Status;
;    0401  2
;    0402  2     BIND
;    0403  2  Comma_Space = $ASCIC (', ') : VECTOR [, BYTE];
;    0404  2
;    0405  2     !+
;    0406  2     ! Initialize Hardware Device Name buffer: point to the first character position.
;    0407  2     !-
;    0408  2
;    0409  2     Char_Ptr = DevNam_String_Buf + 1;
;    0410  2
;    0411  2     !+
;    0412  2     ! Set up for indexing into Test Queue Table
;    0413  2     !-
;    0414  2
;    0415  2     TstQ_Tbl_Index    = 0;
;    0416  2     TstQ_Tbl_Entries = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb_Entries];
;    0417  2     TstQ_Tbl_Ptr      = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr];
;    0418  2
;    0419  2     !+
;    0420  2     ! Get a DDB pointer from the Test Queue Table and determine if the DDB's device failed
;    0421  2     !-
;    0422  2
;    0423  2     WHILE  ((.TstQ_Tbl_Index LSS .TstQ_Tbl_Entries) AND ((DDB_Ptr = .TstQ_Tbl_Ptr [.TstQ_Tbl_Index]) NEQ 0)) DO
;    0424  3  BEGIN
;    0425  3  DDB_Status = .DDB_Ptr [CRD$K_DDB_Status];
;    0426  3
;    0427  3  IF (.CRD$GB_CRD_Debug) THEN
;    0428  4      BEGIN
; P  0429  4      $CRD_Print ($ASCIC ('!/MEN$Print_Failed_Hdwr: DDB_Ptr = !XL(X), DDB_Status = !XL(X)!/'),
;    0430  4   .DDB_Ptr, .DDB_Status);
;    0431  3      END;
;    0432  3
;    0433  4  IF (.DDB_Status <DDB$V_Status_Device_Bad>)
;    0434  3  THEN
;    0435  4      BEGIN
;    0436  4      LOCAL
;    0437  4   Device_Name : REF VECTOR [, BYTE];
;    0438  4
;    0439  4      Numb_Bad_Devices = .Numb Bad Devices + 1;
;    0440  4
```

```
;    0441  4      !+
;    0442  4      ! If DDB's device is indicated failed then load Failed Hardware
;    0443  4      ! buffer with the hardware generic device name
;    0444  4      !-
;    0445  4
;    0446  4      PTable_Ptr    = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    0447  4      ! Point to the P-Table
;    0448  4
;    0449  4      Device_Name = CRD$Get_Device_Name (.PTable_Ptr);
;    0450  4      ! Get a pointer to the ASCIC device name
;    0451  4
;    0452  4      IF (.Numb_Bad_Devices GTR 1) THEN
;    0453  4   Char_Ptr = CH$MOVE (.Comma_Space [0], CH$PTR (Comma_Space [1]), .Char_Ptr);
;    0454  4      ! Move in a ', ' to separate the entries
;    0455  4
;    0456  4      Char_Ptr = CH$MOVE (.Device_Name [0], CH$PTR (Device_Name [1]), .Char_Ptr);
;    0457  4      ! Move in the device name
;    0458  4
;    0459  4      Failed_Hdwr_Found = True;
;    0460  4      ! Indicate at least one failed device was found
;    0461  3      END;
;    0462  3   TstQ_Tbl_Index = .TstQ_Tbl_Index + 1;
;    0463  2   END;
;    0464  2
;    0465  2      !+
;    0466  2      ! If at least one failed device was found then print text
;    0467  2      !-
;    0468  2
;    0469  3      IF (.Failed_Hdwr_Found)
;    0470  2      THEN
;    0471  3   BEGIN
;    0472  3   DevNam_String_Buf [0] = CH$DIFF (.Char_Ptr, CH$PTR (DevNam_String_Buf + 1));
;    0473  3       ! Get string size in bytes
;    0474  3
;    0475  3   $CRD_Message (New_Line, MEN$GT_Failed_Hdwr_Name, DevNam_String_Buf);
;    0476  2   END;
;    0477  2
;    0478  2      RETURN (CRD$K_Success);
;    0479  1 END;



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      20  2C  02  00034 P.AAC:  .ASCII  <2>\, \          ;
61  46  5F  74  6E  69  72  50  24  4E  45  4D  2F  21  40  00037 P.AAD:  .ASCII  \a./MEN$Print_Failed Hdwr: DDB Ptr = !XL(\  ;
5F  42  44  44  20  3A  72  77  64  48  5F  64  65  6C  69  00046            ;
      28  4C  58  21  20  3D  20  72  74  50  00055            ;
20  73  75  74  61  74  53  5F  42  44  44  20  2C  29  58  0005F    .ASCII  \X), DDB_Status =  XL(X) /\       ;
      2F  21  29  58  28  4C  58  21  20  3D  0006E            ;

            COMMA_SPACE=        P.AAC
            .EXTRN  MEN$GT_FAILED_HDWR_NAME

            .PSECT  CODE,NOWRT,  SHR,  PIC,2
```

```
                0FFC 00000     .ENTRY  MEN$PRINT_FAILED_HDWR, Save R2,R3,R4,R5,R6,-; 0349
                R7,R8,R9,R10,R11        ;
                56  7C 00002     CLRQ    FAILED_HDWR_FOUND          ; 0386
                52 00000000'  EF  9E 00004     MOVAB   DEVNAM_STRING_BUF+1, CHAR_PTR        ; 0409
                7E  D4 0000B     CLRL    TSTQ_TBL_INDEX          ; 0415
           00000000G  EF  DD 0000D     PUSHL   MEN$GA_TEST_QUEUE          ; 0416
           00000000G  EF  DD 00013     PUSHL   MEN$GA_TEST_QUEUE+4          ; 0417
                5E  11 00019     BRB     5$              ; 0423
                5A 14   A8  D0 0001B 1$:   MOVL    20(DDB_PTR), DDB_STATUS         ; 0425
                15 00000000G  EF  E9 0001F     BLBC    CRD$GB_CRD_DEBUG, 2$          ; 0427
                0500   8F  BB 00026     PUSHR   #^M<R8,R10>             ; 0430
           00000000'  EF  9F 0002A     PUSHAB  P.AAD           ;
                01  DD 00030     PUSHL   #1              ;
                1A  DD 00032     PUSHL   #26             ;
           00000000G  FF       05  FB 00034     CALLS   #5, @CRD$PRINT          ;
                37        5A       04  E1 0003B 2$:   BBC     #4, DDB_STATUS, 4$          ; 0433
                57  D6 0003F     INCL    NUMB_BAD_DEVICES          ; 0439
                59 08   A8  D0 00041     MOVL    8(DDB_PTR), PTABLE_PTR         ; 0446
                59  DD 00045     PUSHL   PTABLE_PTR          ; 0449
           00000000G  EF       01  FB 00047     CALLS   #1, CRD$GET_DEVICE_NAME         ;
                5B       50  D0 0004E     MOVL    R0, DEVICE_NAME         ;
                01       57  D1 00051     CMPL    NUMB_BAD_DEVICES, #1          ; 0452
                12  15 00054     BLEQ    3$              ;
                50 00000000'  EF  9A 00056     MOVZBL  COMMA_SPACE, R0          ; 0453
                62 00000000'  EF       50  28 0005D     MOVC3   R0, COMMA_SPACE+1, (CHAR_PTR)        ;
                52       53  D0 00065     MOVL    R3, CHAR_PTR            ;
                50       6B  9A 00068 3$:   MOVZBL  (DEVICE_NAME), R0          ; 0456
                62   01    AB       50  28 0006B     MOVC3   R0, 1(DEVICE_NAME), (CHAR_PTR)        ;
                52       53  D0 00070     MOVL    R3, CHAR_PTR            ;
                56       01  D0 00073     MOVL    #1, FAILED_HDWR_FOUND          ; 0459
           08   AE  D6 00076 4$:   INCL    TSTQ_TBL_INDEX          ; 0462
           04   AE 08   AE  D1 00079 5$:   CMPL    TSTQ_TBL_INDEX, TSTQ_TBL_ENTRIES        ; 0423
                0B  18 0007E     BGEQ    6$              ;
                50 08   AE  D0 00080     MOVL    TSTQ_TBL_INDEX, R0          ;
                58 00 BE40  D0 00084     MOVL    @TSTQ_TBL_PTR[R0], DDB_PTR          ;
                90  12 00089     BNEQ    1$              ;
                37       56  E9 0008B 6$:   BLBC    FAILED_HDWR_FOUND, 7$          ; 0469
                50 00000000'  EF  9E 0008E     MOVAB   DEVNAM_STRING_BUF+1, R0          ; 0472
           00000000'  EF       52       50  83 00095     SUBB3   R0, CHAR_PTR, DEVNAM_STRING_BUF        ;
           00000000G  EF  9F 0009D     PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0475
                01  DD 000A3     PUSHL   #1              ;
                1A  DD 000A5     PUSHL   #26             ;
           00000000G  FF       03  FB 000A7     CALLS   #3, @CRD$PRINT          ;
           00000000'  EF  9F 000AE     PUSHAB  DEVNAM_STRING_BUF          ;
           00000000G  EF  9F 000B4     PUSHAB  MEN$GT_FAILED_HDWR_NAME         ;
                01  DD 000BA     PUSHL   #1              ;
                1A  DD 000BC     PUSHL   #26             ;
           00000000G  FF       04  FB 000BE     CALLS   #4, @CRD$PRINT          ;
                50       01  D0 000C5 7$:   MOVL    #1, R0          ; 0478
            04 000C8     RET                ; 0479

; Routine Size:  201 bytes,    Routine Base:  CODE + 019C
```

```
;    0480  1 %SBTTL 'MEN$Print_Runtime_Total Routine'
;    0481  1
;    0482  1 GLOBAL ROUTINE MEN$Print_Runtime_Total =
;    0483  1
;    0484  1 !++
;    0485  1 ! FUNCTIONAL DESCRIPTION:
;    0486  1 !
;    0487  1 !    Prints an ASCII text message of the total test time for the
;    0488  1 !    hardware devices represented in the Test Queue Table.
;    0489  1 !
;    0490  1 ! INPUT PARAMETERS
;    0491  1 !
;    0492  1 ! OUTPUT PARAMETERS
;    0493  1 !
;    0494  1 ! IMPLICIT INPUTS
;    0495  1 !
;    0496  1 !    Test Queue table pointed to by MEN$GA_Test_Queue.
;    0497  1 !
;    0498  1 !    The test time used for each device must be an ASCII representation
;    0499  1 !    of the diagnostic's test time in the following format:
;    0500  1 !
;    0501  1 ! MM:SS
;    0502  1 !
;    0503  1 !      where MM=diagnostic runtime minutes ASCII numeric representation
;    0504  1 !    SS-diagnostic runtime seconds in ASCII numeric representation
;    0505  1 !
;    0506  1 ! EXPLICIT OUTPUTS
;    0507  1 !
;    0508  1 !    FORMAT:
;    0509  1 !
;    0510  1 ! <CRLF><CRLF><Tab><Tab>RUN TIME = MM:SS MINUTES
;    0511  1 !
;    0512  1 !     where MM = Total diagnostic runtime minutes ASCII numeric representation
;    0513  1 !    SS = Total diagnostic runtime seconds numeric representation
;    0514  1 !
;    0515  1 ! SIDE EFFECTS
;    0516  1 !
;    0517  1 !
;    0518  1 ! COMPLETION CODES
;    0519  1 !
;    0520  1 !    CRD$K_Success
;    0521  1 !--
```

```
;    0522  2 BEGIN
;    0523  2     LOCAL
;    0524  2  Runtime_Minutes,
;    0525  2  Runtime_Seconds;
;    0526  2
;    0527  2     REGISTER
;    0528  2    Second_Remaining,
;    0529  2  Minute_Total  : INITIAL (0),
;    0530  2  Second_Total  : INITIAL (0),
;    0531  2  TstQ_Tbl_Index,
;    0532  2  TstQ_Tbl_Entries,
;    0533  2  TstQ_Tbl_Ptr   : REF VECTOR [, LONG],
;    0534  2  HST_No,
;    0535  2  HST_Index,
;    0536  2  Runtime_Ptr,
;    0537  2  DDB_Ptr   : REF Device_Data_Block_Structure,
;    0538  2  PTable_Ptr   : REF $DS_HPO_DECL (),
;    0539  2  HST_Ptr   : REF VECTOR [, LONG];
;    0540  2
;    0541  2     !+
;    0542  2     ! Set up for indexing into Test Queue Table
;    0543  2     !-
;    0544  2
;    0545  2     TstQ_Tbl_Index   = 0;
;    0546  2     TstQ_Tbl_Entries = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb_Entries];
;    0547  2     TstQ_Tbl_Ptr     = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr];
;    0548  2
;    0549  2     !+
;    0550  2     ! Get each DDB pointer from the Test Queue Table
;    0551  2     !-
;    0552  2
;    0553  2    WHILE ((.TstQ_Tbl_Index Lss .TstQ_Tbl_Entries) AND ((DDB_Ptr = .TstQ_Tbl_Ptr [.TstQ_Tbl_Index]) NEQ 0)) DO
;    0554  3 BEGIN
;    0555  3 !+
;    0556  3 ! If Hardware Support table exists then get test times
;    0557  3 !-
;    0558  3
;    0559  4 IF ((HST_Ptr = .DDB_Ptr [CRD$K DDB_Menu_Support_Entry]) NEQ 0)
;    0560  3 THEN
;    0561  4     BEGIN
;    0562  4     !
;    0563  4     ! Setup for indexing thru all concatenated hardware support
;    0564  4     ! tables
;    0565  4     !
;    0566  4
;    0567  4     HST_No     = .DDB_Ptr [CRD$K_DDB_Numb_Support_Tables];
;    0568  4    ! Get number of comcatenated hardware support tables
;    0569  4     HST_Index = - 1;
;    0570  4
;    0571  4     WHILE ((HST_Index = .HST_Index + 1) LSS .HST_No) DO
;    0572  5      BEGIN
;    0573  5   !+
;    0574  5   ! Accumulate test time from all HST's
;    0575  5   !-
;    0576  5
```

```
;    0577  5    Runtime_Minutes = 0;
;    0578  5    Runtime_Seconds = 0;
;    0579  5    Runtime_Ptr     = .HST_Ptr [CRD$K_Runtime];
;    0580  5      ! Point to diagnostic Runtime ASCIC
;    0581  5
;    0582  5      MEN$CVT_Time_ASC_D (.Runtime_Ptr, Runtime_Minutes, Runtime_Seconds);
;    0583  5      ! Get the decimal equivalent of the ASCII runtime
;    0584  5
;    0585  5    Minute_Total = .Minute_Total + .Runtime_Minutes;
;    0586  5      ! Accumulate minutes total
;    0587  5    Second_Total = .Second_Total + .Runtime_Seconds;
;    0588  5      ! Accumulate seconds total
;    0589  5    HST_Ptr      = .HST_Ptr + (CRD$K_Last_Entry * 4);
;    0590  5      ! Next hardware support table
;    0591  4    END;  ! End All HST's
;    0592  3       END;  ! HST exists
;    0593  3
;    0594  3   TstQ_Tbl_Index = .TstQ_Tbl_Index + 1;
;    0595  3      ! next entry
;    0596  2   END;   ! All Test Queue Table Entries
;    0597  2
;    0598  2      !+
;    0599  2      ! Calculate total minutes.
;    0600  2      !-
;    0601  2
;    0602  2      Minute_Total = .Minute_Total + (.Second_Total / 60);
;    0603  2
;    0604  2      !+
;    0605  2      ! Calculate Remaining Seconds
;    0606  2      !-
;    0607  2
;    0608  2      Second Remaining = .Second_Total - ((.Second_Total / 60) * 60);
;    0609  2
;    0610  2      !+
;    0611  2      ! Print total runtime minutes:seconds
;    0612  2      !-
;    0613  2
;    0614  2      $CRD_Print (MEN$GT_Runtime_Total, .Minute_Total, .Second_Remaining);
;    0615  2
;    0616  2      RETURN (CRD$K_Success);
;    0617  1      END;


            .EXTRN  MEN$GT_RUNTIME_TOTAL

              0FFC 00000     .ENTRY  MEN$PRINT_RUNTIME_TOTAL, Save R2,R3,R4,R5,- ; 0482
             R6,R7,R8,R9,R10,R11         ;
            5E       08  C2 00002     SUBL2   #8, SP          ;
            52   7C 00005     CLRQ    MINUTE_TOTAL          ; 0522
            54   D4 00007     CLRL    TSTQ_TBL_INDEX          ; 0545
            55 00000000G   EF  7D 00009     MOVQ    MEN$GA_TEST_QUEUE, TSTQ_TBL_ENTRIES       ; 0546
            36   11 00010     BRB     5$              ; 0553
            5B 10    AA  D0 00012 1$:   MOVL    16(DDB_PTR), HST_PTR          ; 0559
            2E   13 00016     BEQL    4$              ;
```

```
                57 18   AA  D0 00018     MOVL    24(DDB_PTR), HST_NO          ; 0567
                58      01  CE 0001C     MNEGL   #1, HST_INDEX          ; 0569
                1E  11 0001F     BRB     3$              ; 0571
                6E  7C 00021 2$:   CLRQ    RUNTIME_SECONDS          ; 0578
                59 18   AB  D0 00023     MOVL    24(HST_PTR), RUNTIME_PTR         ; 0579
                5E  DD 00027     PUSHL   SP              ; 0582
            08  AE  9F 00029     PUSHAB  RUNTIME_MINUTES          ;
                59  DD 0002C     PUSHL   RUNTIME_PTR             ;
             00000000V  EF       03  FB 0002E     CALLS   #3, MEN$CVT_TIME_ASC_D          ;
                52 04   AE  C0 00035     ADDL2   RUNTIME_MINUTES, MINUTE_TOTAL          ; 0585
                53      6E  C0 00039     ADDL2   RUNTIME_SECONDS, SECOND_TOTAL          ; 0587
                5B      20  C0 0003C     ADDL2   #32, HST_PTR            ; 0589
                58  D6 0003F 3$:   INCL    HST_INDEX           ; 0571
                57      58  D1 00041     CMPL    HST_INDEX, HST_NO          ;
                DB  19 00044     BLSS    2$              ;
                54  D6 00046 4$:   INCL    TSTQ_TBL_INDEX          ; 0594
                55      54  D1 00048 5$:   CMPL    TSTQ_TBL_INDEX, TSTQ_TBL_ENTRIES        ; 0553
                06  18 0004B     BGEQ    6$              ;
                5A    6644  D0 0004D     MOVL    (TSTQ_TBL_PTR)[TSTQ_TBL_INDEX], DDB_PTR      ;
                BF  12 00051     BNEQ    1$              ;
                50      53      3C  C7 00053 6$:   DIVL3   #60, SECOND_TOTAL, R0          ; 0602
                52      50  C0 00057     ADDL2   R0, MINUTE_TOTAL          ;
                50      3C  C4 0005A     MULL2   #60, R0             ; 0608
                50      53      50  C3 0005D     SUBL3   R0, SECOND_TOTAL, SECOND_REMAINING       ;
                50  DD 00061     PUSHL   SECOND_REMAINING          ; 0614
                52  DD 00063     PUSHL   MINUTE_TOTAL            ;
             00000000G  EF  9F 00065     PUSHAB  MEN$GT_RUNTIME_TOTAL          ;
                01  DD 0006B     PUSHL   #1              ;
                1A  DD 0006D     PUSHL   #26             ;
            00000000G  FF       05  FB 0006F     CALLS   #5, @CRD$PRINT          ;
                50      01  D0 00076     MOVL    #1, R0              ; 0616
             04 00079     RET                 ; 0617

; Routine Size:  122 bytes,    Routine Base:  CODE + 0265
```

```
;    0618  1 %SBTTL 'MEN$CVT_Time_ASC_D Routine'
;    0619  1
;    0620  1 GLOBAL ROUTINE MEN$CVT_Time_ASC_D (I_Time, I_Minutes, I_Seconds) =
;    0621  1
;    0622  1 !++
;    0623  1 ! FUNCTIONAL DESCRIPTION:
;    0624  1 !
;    0625  1 !    Converts ASCII representation of minutes and seconds to decimal.
;    0626  1 !
;    0627  1 ! INPUT PARAMETERS
;    0628  1 !
;    0629  1 !    I_TIME  Address of time ASCIC in following format:
;    0630  1 !
;    0631  1 !    mm:ss
;    0632  1 !
;    0633  1 !        where mm=time minutes in ASCII numeric
;    0634  1 !      ss=time seconds in ASCII numeric
;    0635  1 !
;    0636  1 ! OUTPUT PARAMETERS
;    0637  1 !
;    0638  1 !    I_MINUTES  Address of longword to return decimal minutes
;    0639  1 !
;    0640  1 !    I_SECONDS  Address of longword to return decimal seconds
;    0641  1 !
;    0642  1 ! IMPLICIT INPUTS
;    0643  1 !
;    0644  1 ! EXPLICIT OUTPUTS
;    0645  1 !
;    0646  1 ! SIDE EFFECTS
;    0647  1 !
;    0648  1 ! COMPLETION CODES
;    0649  1 !
;    0650  1 !    CRD$K_Success
;    0651  1 !--
```

```
;    0652  2 BEGIN
;    0653  2     REGISTER
;    0654  2   Time_End_Ptr,
;    0655  2   Time_Ptr : REF VECTOR [, BYTE],
;    0656  2   Colon_Ptr,
;    0657  2   Minutes_Ptr,
;    0658  2   Minutes_Sz,
;    0659  2   Seconds_Ptr,
;    0660  2   Seconds_Sz;
;    0661  2
;    0662  2     BIND
;    0663  2   Time    = .I_Time ,
;    0664  2   Minutes = .I_Minutes,
;    0665  2   Seconds = .I_Seconds;
;    0666  2
;    0667  2     Time_Ptr = Time;   ! Point to ASCII time
;    0668  2
;    0669  2     !+
;    0670  2     ! Convert Minutes field
;    0671  2     !-
;    0672  2
;    0673  2     Time_End_Ptr = .Time_Ptr [0] + Time_Ptr [1];
;    0674  2     ! Point to end of ASCII time
;    0675  2     Colon_Ptr    = CH$FIND_CH ((.Time_End_Ptr - Time_Ptr [1]), Time_Ptr [1], %C':');
;    0676  2         ! Point to colon
;    0677  2     Minutes_Ptr  = Time_Ptr [1];
;    0678  2     ! Point to minutes
;    0679  2     Minutes_Sz   = .Colon_Ptr - .Minutes_Ptr;
;    0680  2     ! Size of ASCII minutes field
;    0681  2
;    0682  2     MEN$Scan_Numeric_ASC_D (.Minutes_Sz, .Minutes_Ptr , Minutes);
;    0683  2           ! Convert ASCII minutes to decimal
;    0684  2
;    0685  2     !+
;    0686  2     ! Convert Seconds field
;    0687  2     !-
;    0688  2
;    0689  2     Seconds_Ptr = CH$PLUS (.Colon_Ptr, 1);
;    0690  2     ! Point past colon
;    0691  2     Seconds_Sz  = .Time_End_Ptr - .Seconds_Ptr;
;    0692  2     ! Size of ASCII seconds field
;    0693  2
;    0694  2     MEN$Scan_Numeric_ASC_D (.Seconds_Sz, .Seconds_Ptr , Seconds);
;    0695  2     ! Convert ASCII Seconds to decimal
;    0696  2
;    0697  2     RETURN (CRD$K_Success);
;    0698  1 END;



              001C 00000     .ENTRY  MEN$CVT_TIME_ASC_D, Save R2,R3,R4       ; 0620
          53 04   AC  D0 00002     MOVL    I_TIME, TIME_PTR                 ; 0667
          50      63  9A 00006     MOVZBL  (TIME_PTR), R0                   ; 0673
```

```
                50      53  C0 00009    ADDL2   TIME_PTR, R0                ;
                52 01   A0  9E 0000C    MOVAB   1(R0), TIME_END_PTR         ;
                50        52      53  C3 00010    SUBL3   TIME_PTR, TIME_END_PTR, R0        ; 0675
                51 FF   A0  9E 00014    MOVAB   -1(R0), R1                  ;
                50      53  D0 00018    MOVL    TIME_PTR, R0                ;
           01   A0        51      3A  3A 0001B    LOCC    #58, R1, 1(R0)              ;
                02  12 00020    BNEQ    1$                  ;
                51  D4 00022    CLRL    R1                  ;
                54      51  D0 00024 1$:   MOVL    R1, COLON_PTR               ;
                50 01   A3  9E 00027    MOVAB   1(R3), MINUTES_PTR          ; 0677
                51        54      50  C3 0002B    SUBL3   MINUTES_PTR, COLON_PTR, MINUTES_SZ        ; 0679
           08   AC  DD 0002F    PUSHL   I_MINUTES           ; 0682
                50  DD 00032    PUSHL   MINUTES_PTR           ;
                51  DD 00034    PUSHL   MINUTES_SZ            ;
            00000000V  EF       03  FB 00036    CALLS   #3, MEN$SCAN_NUMERIC_ASC_D        ;
                50 01   A4  9E 0003D    MOVAB   1(R4), SECONDS_PTR          ; 0689
                51        52      50  C3 00041    SUBL3   SECONDS_PTR, TIME_END_PTR, SECONDS_SZ        ; 0691
           0C   AC  DD 00045    PUSHL   I_SECONDS           ; 0694
                50  DD 00048    PUSHL   SECONDS_PTR           ;
                51  DD 0004A    PUSHL   SECONDS_SZ            ;
            00000000V  EF       03  FB 0004C    CALLS   #3, MEN$SCAN_NUMERIC_ASC_D        ;
                50      01  D0 00053    MOVL    #1, R0              ; 0697
             04 00056    RET                 ; 0698

; Routine Size:  87 bytes,    Routine Base:  CODE + 02DF
```

```
;    0699  1 %SBTTL 'MEN$Scan_Numeric_ASC_D Routine'
;    0700  1
;    0701  1 GLOBAL ROUTINE MEN$Scan_Numeric_ASC_D (I_Length, I_Pointer, I_RetVal) =
;    0702  1
;    0703  1 !++
;    0704  1 ! FUNCTIONAL DESCRIPTION:
;    0705  1 !
;    0706  1 !    Interface routine to VSD Scan_Numeric Routine.  Converts ASCII representation
;    0707  1 !    of decimal number to decimal value specified.  The reason for this routine is
;    0708  1 !    because the VDS routine does not preserve registers.  This interface routine
;    0709  1 !    will guarantee that the calling routine registers 2-11 will be preserved.
;    0710  1 !
;    0711  1 ! INPUT PARAMETERS
;    0712  1 !
;    0713  1 !    I_Length  Size of ASCII  string
;    0714  1 !
;    0715  1 !    I_Pointer  Address of ASCII  string
;    0716  1 !
;    0717  1 ! OUTPUT PARAMETERS
;    0718  1 !
;    0719  1 !    I_RetVal  Address of longword to return value
;    0720  1 !
;    0721  1 ! IMPLICIT INPUTS
;    0722  1 !
;    0723  1 ! EXPLICIT OUTPUTS
;    0724  1 !
;    0725  1 ! SIDE EFFECTS
;    0726  1 !
;    0727  1 ! COMPLETION CODES
;    0728  1 !
;    0729  1 !    CRD$K_Success
;    0730  1 !--
```

```
;    0731  2 BEGIN
;    0732  2     BIND ROUTINE
;    0733  2  Scan_Numeric = (.CRD$Scan$Numeric) : JSB_Scan_Numeric;
;    0734  2
;    0735  2     BIND
;    0736  2  Length  = .I_Length,
;    0737  2  Pointer = .I_Pointer,
;    0738  2  RetVal  = .I_RetVal;
;    0739  2
;    0740  2     Scan_Numeric (10, Length, Pointer ; RetVal);
;    0741  2
;    0742  2     RETURN (CRD$K_Success);
;    0743  1 END;



            .EXTRN  CRD$SCAN$NUMERIC

              0FFC 00000     .ENTRY  MEN$SCAN_NUMERIC_ASC_D, Save R2,R3,R4,R5,-  ; 0701
             R6,R7,R8,R9,R10,R11         ;
            54 04    AC  7D 00002     MOVQ    I_LENGTH, R4          ; 0740
            53       0A  D0 00006     MOVL    #10, R3          ;
         00000000G   FF  16 00009     JSB     @CRD$SCAN$NUMERIC          ;
        0C   BC      50  D0 0000F     MOVL    R0, @I_RETVAL          ;
            50       01  D0 00013     MOVL    #1, R0          ; 0742
          04 00016     RET                 ; 0743

; Routine Size:  23 bytes,    Routine Base:  CODE + 0336
```

```
;    0744  1 %SBTTL 'MEN$Preparation_Err_Text Routine'
;    0745  1
;    0746  1 GLOBAL ROUTINE MEN$Preparation_Err_text (User_Text_Ptr, DDB_Ptr) =
;    0747  1
;    0748  1 !++
;    0749  1 ! FUNCTIONAL DESCRIPTION:
;    0750  1 !
;    0751  1 !    Prints text associated with a Hardware Test Preparation error.
;    0752  1 !
;    0753  1 ! INPUT PARAMETERS
;    0754  1 !
;    0755  1 !    User_Text_Ptr: Address of ASCIC Text associated with hardware test
;    0756  1 !   preparation error. This address COULD be 0 though.     ![01]
;    0757  1 !
;    0758  1 !    DDB_Ptr:  Address of DDB associated with hardware device
;    0759  1 !    from which a hardware preparation error has occured
;    0760  1 ! OUTPUT PARAMETERS
;    0761  1 !
;    0762  1 ! IMPLICIT INPUTS
;    0763  1 !
;    0764  1 ! EXPLICIT OUTPUTS
;    0765  1 !
;    0766  1 !    Formatted ASCII printout at terminal with format:
;    0767  1 !
;    0768  1 !  <CR/LF><CR/LF>
;    0769  1 !  ** 'Text message pointed to by User_Text_Ptr' **
;    0770  1 !  <CR/LF>
;    0771  1 !  ** 'Generic Device Name' - INCORRECT HARDWARE TEST PREPARATION **
;    0772  1 !  <CR/LF>
;    0773  1 !
;    0774  1 ! SIDE EFFECTS
;    0775  1 !
;    0776  1 ! COMPLETION CODES
;    0777  1 !
;    0778  1 !    CRD$K_Success
;    0779  1 !--
```

```
;    0780  2 BEGIN
;    0781  2  MAP
;    0782  2   User_Text_Ptr : REF VECTOR [, BYTE],
;    0783  2   DDB_Ptr       : REF Device_Data_Block_Structure;
;    0784  2
;    0785  2  REGISTER
;    0786  2   Device_Name_Ptr : REF VECTOR [, BYTE];
;    0787  2
;    0788  2  Device_Name_Ptr = CRD$Get_Device_Name (.DDB_Ptr [CRD$K_DDB_PTable_Entry]);
;    0789  2
;    0790  2  !+                 ![01]
;    0791  2  ! Check first to see if there is an address given by the user for the error message. That is,  ![01]
;    0792  2  ! is the MSGADR parameter equal to 0. If so, print a short message.     ![01]
;    0793  2  !-                 ![01]
;    0794  2
;    0795  2  IF (.User_Text_Ptr EQL 0) THEN
;    0796  3   $CRD_Print (MEN$GT_Prep_Error_Text_No_User, .Device_Name_Ptr)
;    0797  3
;    0798  3
;    0799  3  !+                 ![01]
;    0800  3  ! The user actually passed a valid address of a non-null string. Print it.    ![01]
;    0801  3  !-                 ![01]
;    0802  3
;    0803  2  ELSE
;    0804  3   BEGIN
;    0805  3   MAP
;    0806  3    MEN$GT_Prep_Error_Text : VECTOR [, BYTE];
;    0807  3
;    0808  3   REGISTER
;    0809  3    Left,
;    0810  3    Right,
;    0811  3    Prep_Text_Len,
;    0812  3    User_Text_Len,
;    0813  3    Difference;
;    0814  3
;    0815  3 !  Prep_Text_Len = .Device_Name_Ptr [0] + MEN$K_Prep_Error_Text_Len;
;    0816  3 !  User_Text_Len = .User_Text_Ptr [0];
;    0817  3
;    0818  3 !  Difference = ABS (.User_Text_Len - .Prep_Text_Len);
;    0819  3
;    0820  3 !  Left  = (.Difference + 1) / 2;
;    0821  3 !  Right = .Difference - .Left;
;    0822  3   Left=1;             ![02]
;    0823  3   Right=1;            ![02]
;    0824  3
;    0825  3 !  IF (.User_Text_Len GTR .Prep_Text_Len) THEN
;    0826  3 !   $CRD_Print (MEN$GT_Prep_Error_Text,
;    0827  3 !    0,     User_Text_Ptr [0],   0,
;    0828  3 !    .Left, Device_Name_Ptr [0], .Right)
;    0829  3 !  ELSE
; P  0830  3    $CRD_Print (MEN$GT_Prep_Error_Text,
; P  0831  3     .Left, User_Text_Ptr [0],   .Right,
;    0832  3     0,     Device_Name_Ptr [0], 0);
;    0833  2   END;
;    0834  2     RETURN (CRD$K_Success);
```

```
;   0835  1 END;


        .EXTRN  MEN$GT_PREP_ERROR_TEXT_NO_USER
        .EXTRN  MEN$GT_PREP_ERROR_TEXT

          0004 00000     .ENTRY  MEN$PREPARATION_ERR_TEXT, Save R2       ; 0746
       50 08    AC  D0 00002    MOVL    DDB_PTR, R0                     ; 0788
    08 A0  DD 00006    PUSHL   8(R0)                   ;
     00000000G  EF       01  FB 00009    CALLS   #1, CRD$GET_DEVICE_NAME         ;
       52        50  D0 00010    MOVL    R0, DEVICE_NAME_PTR             ;
    04 AC  D5 00013    TSTL    USER_TEXT_PTR                   ; 0795
       15  12 00016    BNEQ    1$                      ;
       52  DD 00018    PUSHL   DEVICE_NAME_PTR                 ; 0796
     00000000G  EF  9F 0001A    PUSHAB  MEN$GT_PREP_ERROR_TEXT_NO_USER          ;
       01  DD 00020    PUSHL   #1                      ;
       1A  DD 00022    PUSHL   #26                     ;
     00000000G  FF       04  FB 00024    CALLS   #4, @CRD$PRINT          ;
       24  11 0002B    BRB     2$                      ;
       50        01  D0 0002D 1$:   MOVL    #1, LEFT                        ; 0822
       51        01  D0 00030    MOVL    #1, RIGHT                       ; 0823
       7E  D4 00033    CLRL    -(SP)                   ; 0832
       52  DD 00035    PUSHL   DEVICE_NAME_PTR                 ;
       7E  D4 00037    CLRL    -(SP)                   ;
       51  DD 00039    PUSHL   RIGHT                   ;
    04 AC  DD 0003B    PUSHL   USER_TEXT_PTR                   ;
       50  DD 0003E    PUSHL   LEFT                    ;
     00000000G  EF  9F 00040    PUSHAB  MEN$GT_PREP_ERROR_TEXT          ;
       01  DD 00046    PUSHL   #1                      ;
       1A  DD 00048    PUSHL   #26                     ;
     00000000G  FF       09  FB 0004A    CALLS   #9, @CRD$PRINT          ;
       50        01  D0 00051 2$:   MOVL    #1, R0                  ; 0834
        04 00054    RET                     ; 0835

; Routine Size:  85 bytes,    Routine Base:  CODE + 034D
```

```
;    0836  1 %SBTTL 'MEN$Scratch_Media_Msg Routine'
;    0837  1
;    0838  1 GLOBAL ROUTINE MEN$Scratch_Media_Msg  =
;    0839  1
;    0840  1 !++
;    0841  1 ! FUNCTIONAL DESCRIPTION:
;    0842  1 !
;    0843  1 !    Prints a warning message and a list of hardware names
;    0844  1 !    indicating that scratch media must be provided
;    0845  1 !    for the hardware selected for test.
;    0846  1 !    Only hardware whose test configuration block indicates that the hardware
;    0847  1 !    requires scratch media is considered for printout.
;    0848  1 !
;    0849  1 !
;    0850  1 ! INPUT PARAMETERS
;    0851  1 !
;    0852  1 ! OUTPUT PARAMETERS
;    0853  1 !
;    0854  1 ! IMPLICIT INPUTS
;    0855  1 !
;    0856  1 !    Test Queue table pointed to by MEN$GA_Test_Queue.
;    0857  1 !
;    0858  1 ! EXPLICIT OUTPUTS
;    0859  1 !
;    0860  1 !    ASCII text printout.  A pause occurs and the operator is prompted
;    0861  1 !    to press carriage return to continue.
;    0862  1 !
;    0863  1 !
;    0864  1 ! SIDE EFFECTS
;    0865  1 !
;    0866  1 !
;    0867  1 ! COMPLETION CODES
;    0868  1 !
;    0869  1 !     CRD$K Success Printout occurred - at least one hardware device
;    0870  1 !    was found to require scratch media.
;    0871  1 !
;    0872  1 !    No printout occurred - no hardware was specified
;    0873  1 !    to need scratch media.
;    0874  1 !
```

```
;    0875  2 BEGIN
;    0876  2     LOCAL
;    0877  2  TstQ_Tbl_Index,
;    0878  2  TstQ_Tbl_Entries,
;    0879  2  TstQ_Tbl_Ptr  : REF VECTOR [, LONG];
;    0880  2
;    0881  2     REGISTER
;    0882  2  Valid_Status,
;    0883  2  Hdwr_Found  : INITIAL (False), ! Asume no hardware found
;    0884  2  Printed_Basic_Msg : INITIAL (False), ! Assume basic message not printed
;    0885  2  Hdwr_Name_Ptr,
;    0886  2  Dev_Name,
;    0887  2
;    0888  2  DDB_Ptr   : REF Device_Data_Block_Structure,
;    0889  2  PTable_Ptr  : REF $DS_HPO_DECL (),
;    0890  2  SupBlk_Ptr  : REF $MEN_SupBlk_Attr,
;    0891  2  TstCnfg_Ptr  : REF $MEN_TstCnfg_ATTR;
;    0892  2
;    0893  2     BIND
;    0894  2  T_Prompt_Default = $ASCIC ('CRLF') : VECTOR [,BYTE];
;    0895  2
;    0896  2     !+
;    0897  2     ! Set up for indexing into Test Queue Table
;    0898  2     !-
;    0899  2
;    0900  2     TstQ_Tbl_Index   = 0;
;    0901  2     TstQ_Tbl_Entries = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb_Entries];
;    0902  2     TstQ_Tbl_Ptr     = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr];
;    0903  2
;    0904  2     !+
;    0905  2     ! Get all Hardware Names from the DDB's in the Test Queue Table whose
;    0906  2     ! hardware test configuration block indicates that scratch media
;    0907  2     ! is required.
;    0908  2     !-
;    0909  2
;    0910  2     WHILE ((.TstQ_Tbl_Index LSS .TstQ_Tbl_Entries) AND ((DDB_Ptr = .TstQ_Tbl_Ptr [.TstQ_Tbl_Index]) NEQ 0)) DO
;    0911  3  BEGIN
;    0912  3  SupBlk_Ptr  = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr];
;    0913  3     ! Get support block
;    0914  3  TstCnfg_Ptr = SupBlk_Ptr [SupBlk$B_StrtFrst_CnfgBlk];
;    0915  3     ! Get hardware configuration
;    0916  3
;    0917  3  !+
;    0918  3  ! If scratch media is required print the list of hardware
;    0919  3  !-
;    0920  3
;    0921  4  IF (.TstCnfg_Ptr [TstCnfg$V_Status_Scratch])
;    0922  3  THEN
;    0923  4      BEGIN
;    0924  4      !+
;    0925  4      ! Print the basic message only once
;    0926  4      !-
;    0927  4
;    0928  5      IF (NOT .Printed_Basic_Msg)
;    0929  4      THEN
```

```
;    0930  5   BEGIN
;    0931  5   $CRD_Print (MEN$GT_ScrMedia_Msg);
;    0932  5   Printed_Basic_Msg = True;
;    0933  5   Hdwr_Found        = True;
;    0934  5       ! Indicate at least one hdwr found
;    0935  4   END;
;    0936  4
;    0937  4       !+
;    0938  4       ! Set up for printing Hardware Name
;    0939  4       !-
;    0940  4
;    0941  4       PTable_Ptr    = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    0942  4      ! Point to the P-Table
;    0943  4       Hdwr_Name_Ptr = PTable_Ptr [HP$T_TYPE];
;    0944  4      ! Point to the Hardware device name ASCIC
;    0945  4
;    0946  4       !+
;    0947  4       ! Set up for printing Generic Device Name
;    0948  4       !-
;    0949  4
;    0950  4       Dev_Name = CRD$Get_Device_Name (.PTable_Ptr);
;    0951  4      ! Get the generic device name ASCIC
;    0952  4
;    0953  4       !+
;    0954  4       ! Print the hardware text
;    0955  4       !-
;    0956  4
; P  0957  4       $CRD_Print (MEN$GT_ScrMedia_Hdwr,
; P  0958  4       .Hdwr_Name_Ptr, ! Hardware Name ASCIC
; P  0959  4       .Dev_Name  ! Generic Device Name ASCIC
;    0960  4    );
;    0961  3       END;    ! Printing list of hardware
;    0962  3
;    0963  3   TstQ_Tbl_Index = .TstQ_Tbl_Index + 1;
;    0964  3       ! Advance to the next entry in the Test Queue Table
;    0965  2   END;     ! Indexing into Test Queue Table
;    0966  2
;    0967  2      !+
;    0968  2      ! If no hardware name was found then return. If one was found, print the last line of the message.
;    0969  2      !-
;    0970  2
;    0971  3      IF (NOT .Hdwr_Found)
;    0972  2      THEN
;    0973  3   RETURN (CRD$K_Success)
;    0974  2      ELSE
;    0975  2   $CRD_Print (MEN$GT_ScrMedia_Hdwr_End);
;    0976  2
;    0977  2      !+
;    0978  2      ! Pause and prompt operator to press RETURN to continue
;    0979  2      !-
;    0980  2
;    0981  2      Valid_Status = False;  ! Assume the worst
;    0982  2
;    0983  2      DO
; P  0984  3   IF ($MEN_ASKSTR (Msg Adr = MEN$GT_ScrMedia_Prompt,
```

```
; P 0985  3     Buf_Adr = MEN$GT_OperInput_Buffer,
;   0986  3     Def_Adr = T_Prompt_Default))
;   0987  2  THEN
;   0988  2       !+
;   0989  2          ! Valid response to prompt is either <CR/LF> or ^C followed by the 'Resume' choice
;   0990  2       ! from the ^C Menu. In both cases the default is deposited in the operator buffer.
;   0991  2       !-
;   0992  2
;   0993  5       IF (NOT (Valid_Status = (CH$EQL (.MEN$GT_OperInput_Buffer [0], MEN$GT_OperInput_Buffer [1],
;   0994  3            .T_Prompt_Default [0],    T_Prompt_Default [1]))))
;   0995  2       THEN
;   0996  3    $CRD_Message (New_Line, MEN$GT_InvOperInput)
;   0997  2       UNTIL (.Valid_Status);  ! Repeat this till we get a valid answer
;   0998  2
;   0999  2     RETURN (CRD$K_Success);
;   1000  1 END;


          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

     46  4C  52  43  04  00078 P.AAE:  .ASCII  <4>\CRLF\          ;

         T_PROMPT_DEFAULT=   P.AAE
          .EXTRN  MEN$GT_SCRMEDIA_MSG
          .EXTRN  MEN$GT_SCRMEDIA_HDWR
          .EXTRN  MEN$GT_SCRMEDIA_HDWR_END
          .EXTRN  MEN$GT_SCRMEDIA_PROMPT
          .EXTRN  MEN$GT_OPERINPUT_BUFFER
          .EXTRN  DS$BREAK, DS$ASKSTR
          .EXTRN  MEN$GT_INVOPERINPUT

          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            0FFC 00000     .ENTRY  MEN$SCRATCH_MEDIA_MSG, Save R2,R3,R4,R5,R6,-; 0838
           R7,R8,R9,R10,R11         ;
          52  7C 00002     CLRQ    HDWR_FOUND          ; 0875
          5A  D4 00004     CLRL    TSTQ_TBL_INDEX          ; 0900
00000000G  EF  DD 00006     PUSHL   MEN$GA_TEST_QUEUE         ; 0901
          5B 00000000G  EF  D0 0000C     MOVL    MEN$GA_TEST_QUEUE+4, TSTQ_TBL_PTR       ; 0902
          4F  11 00013     BRB     4$              ; 0910
          58 20   A6  D0 00015 1$:   MOVL    32(DDB_PTR), SUPBLK_PTR         ; 0912
          59 17   A8  9E 00019     MOVAB   23(R8), TSTCNFG_PTR          ; 0914
          41 02   A9  E9 0001D     BLBC    2(TSTCNFG_PTR), 3$          ; 0921
          17      53  E8 00021     BLBS    PRINTED_BASIC_MSG, 2$           ; 0928
00000000G  EF  9F 00024     PUSHAB  MEN$GT_SCRMEDIA_MSG         ; 0931
          01  DD 0002A     PUSHL   #1              ;
          1A  DD 0002C     PUSHL   #26             ;
00000000G  FF      03  FB 0002E     CALLS   #3, @CRD$PRINT          ;
          53      01  D0 00035     MOVL    #1, PRINTED_BASIC_MSG           ; 0932
          52      01  D0 00038     MOVL    #1, HDWR_FOUND          ; 0933
          57 08   A6  D0 0003B 2$:   MOVL    8(DDB_PTR), PTABLE_PTR          ; 0941
          54 26   A7  9E 0003F     MOVAB   38(R7), HDWR_NAME_PTR           ; 0943
          57  DD 00043     PUSHL   PTABLE_PTR          ; 0950
00000000G  EF      01  FB 00045     CALLS   #1, CRD$GET_DEVICE_NAME         ;
```

```
                55       50  D0 0004C     MOVL    R0, DEV_NAME            ;
                30  BB 0004F     PUSHR   #^M<R4,R5>              ; 0960
          00000000G  EF  9F 00051     PUSHAB  MEN$GT_SCRMEDIA_HDWR            ;
                01  DD 00057     PUSHL   #1                ;
                1A  DD 00059     PUSHL   #26                ;
          00000000G  FF       05  FB 0005B     CALLS   #5, @CRD$PRINT          ;
                5A  D6 00062 3$:    INCL    TSTQ_TBL_INDEX          ; 0963
                6E       5A  D1 00064 4$:    CMPL    TSTQ_TBL_INDEX, TSTQ_TBL_ENTRIES        ; 0910
                06  18 00067     BGEQ    5$                ;
                56     6B4A  D0 00069     MOVL    (TSTQ_TBL_PTR)[TSTQ_TBL_INDEX], DDB_PTR     ;
                A6  12 0006D     BNEQ    1$                ;
                03       52  E8 0006F 5$:    BLBS    HDWR_FOUND, 6$          ; 0971
              0098  31 00072     BRW     10$               ;
          00000000G  EF  9F 00075 6$:    PUSHAB  MEN$GT_SCRMEDIA_HDWR_END        ; 0975
                01  DD 0007B     PUSHL   #1                ;
                1A  DD 0007D     PUSHL   #26                ;
          00000000G  FF       03  FB 0007F     CALLS   #3, @CRD$PRINT          ;
                54  D4 00086     CLRL    VALID_STATUS            ; 0981
          00000000G  9F       00  FB 00088 7$:    CALLS   #0, @#DS$BREAK          ; 0986
                7E  D4 0008F     CLRL    -(SP)             ;
          00000000'  EF  9F 00091     PUSHAB  T_PROMPT_DEFAULT          ;
                7E  D4 00097     CLRL    -(SP)             ;
                7E 48   8F  9A 00099     MOVZBL  #72, -(SP)              ;
          00000000G  EF  9F 0009D     PUSHAB  MEN$GT_OPERINPUT_BUFFER         ;
          00000000G  EF  9F 000A3     PUSHAB  MEN$GT_SCRMEDIA_PROMPT          ;
          00000000G  9F       06  FB 000A9     CALLS   #6, @#DS$ASKSTR         ;
                52       50  D0 000B0     MOVL    R0, STATUS              ;
          00000000G  9F       00  FB 000B3     CALLS   #0, @#DS$BREAK          ;
                4A       52  E9 000BA     BLBC    STATUS, 9$              ;
                51 00000000G  EF  9A 000BD     MOVZBL  MEN$GT_OPERINPUT_BUFFER, R1         ; 0993
                50 00000000'  EF  9A 000C4     MOVZBL  T_PROMPT_DEFAULT, R0            ; 0994
                55  D4 000CB     CLRL    R5                ;
           50           00 00000000G  EF       51  2D 000CD     CMPC5   R1, MEN$GT_OPERINPUT_BUFFER+1, #0, R0, -     ;
              00000000'  EF       000D6     T_PROMPT_DEFAULT+1          ;
                02  12 000DB     BNEQ    8$                ;
                55  D6 000DD     INCL    R5                ;
                54       55  D0 000DF 8$:    MOVL    R5, VALID_STATUS          ; 0993
                22       55  E8 000E2     BLBS    R5, 9$                  ;
          00000000G  EF  9F 000E5     PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0996
                01  DD 000EB     PUSHL   #1                ;
                1A  DD 000ED     PUSHL   #26                ;
          00000000G  FF       03  FB 000EF     CALLS   #3, @CRD$PRINT          ;
          00000000G  EF  9F 000F6     PUSHAB  MEN$GT_INVOPERINPUT           ;
                01  DD 000FC     PUSHL   #1                ;
                1A  DD 000FE     PUSHL   #26                ;
          00000000G  FF       03  FB 00100     CALLS   #3, @CRD$PRINT          ;
                03       54  E8 00107 9$:    BLBS    VALID_STATUS, 10$          ; 0997
              FF7B  31 0010A     BRW     7$                ;
                50       01  D0 0010D 10$:    MOVL    #1, R0          ; 0999
               04 00110     RET                     ; 1000

; Routine Size:  273 bytes,    Routine Base:  CODE + 03A2
```

```
;    1001  1 %SBTTL 'MEN$Menu_Summary'
;    1002  1
;    1003  1 GLOBAL ROUTINE MEN$Menu_Summary : NOVALUE =
;    1004  1
;    1005  1 !++
;    1006  1 ! FUNCTIONAL DESCRIPTION:
;    1007  1 !
;    1008  1 ! Prints summary of devices tested after a
;    1009  1 !  Main Menu choice of E (Test all devices)
;    1010  1 ! or
;    1011  1 !  Main Menu choice of D (Test Menu of devices) followed by
;    1012  1 !  Test Menu choice of (Test all devices)
;    1013  1 !
;    1014  1 ! INPUT PARAMETERS
;    1015  1 !
;    1016  1 ! None
;    1017  1 !
;    1018  1 ! OUTPUT PARAMETERS
;    1019  1 !
;    1020  1 ! None
;    1021  1 !
;    1022  1 ! SIDE EFFECTS
;    1023  1 !
;    1024  1 ! None
;    1025  1 !
;    1026  1 ! COMPLETION CODES
;    1027  1 !
;    1028  1 ! None
;    1029  1 !--
```

```
;    1030  2 BEGIN     ! Start of MEN$Menu_Summary routine
;    1031  2
;    1032  2     REGISTER
;    1033  2  TstQ_Tbl_Index,
;    1034  2  TstQ_Tbl_Entries,
;    1035  2  TstQ_Tbl_Ptr : REF VECTOR [, LONG],
;    1036  2  DDB_Ptr  : REF Device_data_Block_Structure,
;    1037  2  SupBlk_Ptr : REF $MEN_SupBlk_ATTR,
;    1038  2  PTable_Ptr : REF $DS_HPO_DECL ();
;    1039  2
;    1040  2     LOCAL
;    1041  2  Some_Fail: INITIAL (False),
;    1042  2  Some_Pass: INITIAL (False),
;    1043  2  Some_Inc_Prep: INITIAL (False),
;    1044  2  !
;    1045  2  ! NOTE: there is no need for Some_Other because the menu does not
;    1046  2  ! consider devices that are not available.
;    1047  2  !
;    1048  2  Hdwr_Name_Ptr,
;    1049  2  Dev_Name,
;    1050  2  Console_Char : $CRD_Termchar_ATTR,
;    1051  2  Screen_Counter;
;    1052  2
;    1053  2
;    1054  2     !+
;    1055  2     ! BEFORE DOING ANYTHING, CHECK THAT THE CONSOLE IS SOFT (i.e. VT100 or VT52)
;    1056  2     ! AND NOT HARD (i.e. LA34 etc.)
;    1057  2     !-
;    1058  3     IF ($DS_GETTERM (TERMCHAR = Console_Char))
;    1059  3      ! If SS$_NORMAL status on getting
;    1060  3      ! terminal characteristics
;    1061  2     THEN
;    1062  3       BEGIN
;    1063  3       !+
;    1064  3  ! If console is not a CRD supported soft console, then just RETURN.
;    1065  3  ! Otherwise, continue normally.
;    1066  3  !-
;    1067  4  IF ((.Console_Char [Termchar$B_Type] NEQ TT$_VT100) AND
;    1068  4 !***     (.Console_Char [Termchar$B_Type] NEQ TT$_VS125) AND     ![04]
;    1069  5      (.Console_Char [Termchar$B_Type] NEQ TT$_VT52)
;    1070  4     )
;    1071  3  THEN
;    1072  3   RETURN;
;    1073  3       END
;    1074  2     ELSE
;    1075  2      $CRD_Print (CRD$GT_Internal_Error);
;    1076  2
;    1077  2
;    1078  2     !+
;    1079  2     ! Print the Summary Title
;    1080  2     !-
;    1081  2     $CRD_Print (CRD$GT_Summary_Title);
;    1082  2
;    1083  2
;    1084  2
```

```
;    1085  2     INCR I FROM 0 TO 3 DO
;    1086  3  BEGIN    ! Begin of INCR I
;    1087  3  !+
;    1088  3  ! Print the Subtitle of the Auto Test Summary
;    1089  3  !-
;    1090  3  SELECTONE .I OF
;    1091  3   SET
;    1092  3   [1]: IF (.Some_Fail) THEN $CRD_Print (CRD$GT_All_Failures);
;    1093  3   [2]: IF (.Some_Pass) THEN $CRD_Print (CRD$GT_All_Passes);
;    1094  3   [3]: IF (.Some_Inc_Prep) THEN $CRD_Print (CRD$GT_All_Inc_Hdwr_Prep);
;    1095  3   TES;
;    1096  3
;    1097  3  !+
;    1098  3  ! Set up for indexing into Test Queue Table
;    1099  3  !-
;    1100  3
;    1101  3  TstQ_Tbl_Index   = 0;
;    1102  3  TstQ_Tbl_Entries = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb_Entries];
;    1103  3  TstQ_Tbl_Ptr     = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr];
;    1104  3
;    1105  3
;    1106  3  Screen_Counter = 0;  ! Init number of entries actually on
;    1107  3      ! the screen. Max numb is 2.
;    1108  3
;    1109  3  !+
;    1110  3  ! The following WHILE-DO acts on each DDB in the DDBs-queue.
;    1111  3  ! Get the DDB pointers from the Test Queue Table.
;    1112  3  !-
;    1113  3
;    1114  3  WHILE (.TstQ_Tbl_Index LSS .TstQ_Tbl_Entries) AND
;    1115  4   ((DDB_Ptr = .TstQ_Tbl_Ptr [.TstQ_Tbl_Index]) NEQ 0)
;    1116  3       DO
;    1117  3   !+
;    1118  3   ! This is a large DO in the WHILE-DO command.
;    1119  3   !-
;    1120  4   BEGIN    ! Begin of DO command.
;    1121  4
;    1122  4   REGISTER
;    1123  4    Prep_Error : BYTE,
;    1124  4    Bad    : BYTE,
;    1125  4    Finished : BYTE,
;    1126  4    DDB_Fail : BYTE,
;    1127  4    DDB_Pass : BYTE,
;    1128  4    DDB_Inc_Prep : BYTE;
;    1129  4
;    1130  4   !+
;    1131  4   ! Note that every device in the MENU portion is considered
;    1132  4   ! ESSENTIAL so that the DDB essential bit is not checked.
;    1133  4   !-
;    1134  4
;    1135  4   Finished  = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Dev_Test_Complete>;
;    1136  4   Bad    = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Device_Bad>;
;    1137  4   Prep_Error  = .DDB_Ptr [CRD$K_DDB_Status] <DDB$V_Status_Preparation_Error>;
;    1138  4
;    1139  4   !+
```

```
;    1140  4    ! Determine if DDB is FAIL
;    1141  4    !-
;    1142  6    DDB_Fail = ( ((NOT(.Finished)) OR .Bad)
;    1143  5      AND
;    1144  6      (NOT (.Prep_Error))
;    1145  4           );
;    1146  4
;    1147  4    !+
;    1148  4    ! Determine if Incorrect Preparation
;    1149  4    !-
;    1150  5    DDB_Inc_Prep = (
;    1151  6             Boolean (.Prep_Error)
;    1152  4          );
;    1153  4
;    1154  4    !+
;    1155  4    ! Determine if DDB is PASS
;    1156  4    !-
;    1157  4    DDB_Pass = ((NOT(.DDB_Fail))  AND  (NOT(.DDB_Inc_Prep)));
;    1158  4
;    1159  4    Some_Fail = .Some_Fail OR .DDB_Fail;
;    1160  4    Some_Pass = .Some_Pass OR .DDB_Pass;
;    1161  4    Some_Inc_Prep = .Some_Inc_Prep OR .DDB_Inc_Prep;
;    1162  4
;    1163  5    IF (SELECTONE .I OF
;    1164  5     SET
;    1165  5     [0]: False;
;    1166  5     [1]: .DDB_Fail;
;    1167  5     [2]: .DDB_Pass;
;    1168  5     [3]: .DDB_Inc_Prep;
;    1169  5     TES
;    1170  5       )    ! End of SELECTONE statement
;    1171  4    THEN
;    1172  5         BEGIN
;    1173  5     !+
;    1174  5     ! Set up for printing Hardware Name
;    1175  5     !-
;    1176  5
;    1177  5     PTable_Ptr = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    1178  5       ! Point to the P-Table
;    1179  5     Hdwr_Name_Ptr = PTable_Ptr [HP$T_TYPE];
;    1180  5       ! Point to the Hardware device name ASCIC
;    1181  5
;    1182  5     !+
;    1183  5     ! Set up for printing Generic Device Name
;    1184  5     !-
;    1185  5
;    1186  5     Dev_Name = CRD$Get_Device_Name (.PTable_Ptr);
;    1187  5       ! Get an ASCIC device name string from the Ptable
;    1188  5
;    1189  5
;    1190  5     !+
;    1191  5     ! Print the system hardware text
;    1192  5     !-
;    1193  5
;    1194  6     IF (.Screen_Counter EQL 3)
```

```
;    1195  5     THEN
;    1196  6      BEGIN
;    1197  6      $CRD_Print (CRD$GT_New_Line);
;    1198  6      Screen_Counter = 0;
;    1199  5      END;
;    1200  5
;    1201  5     $CRD_Print (CRD$GT_Tab);
;    1202  5     $CRD_Print (.Hdwr_Name_Ptr);
;    1203  5     $CRD_Print (CRD$GT_Slash);
;    1204  5     $CRD_Print (.Dev_Name);
;    1205  5     $CRD_Print (CRD$GT_Tab);
;    1206  5
;    1207  5     Screen_Counter = .Screen_Counter + 1;
;    1208  5
;    1209  5         END
;    1210  4    ;    ! End of large IF THEN command
;    1211  4
;    1212  4
;    1213  4    !+
;    1214  4    ! We are done processing this DDB so move on to the next DDB
;    1215  4    !-
;    1216  4
;    1217  4    TstQ_Tbl_Index  = .TstQ_Tbl_Index + 1;
;    1218  4
;    1219  3    END;    ! End of DO command in WHILE-DO loop.
;    1220  3
;    1221  3
;    1222  4      IF (.I EQL 3)    ! Make sure there is a space between
;    1223  3      THEN     ! the end of summary the next message.
;    1224  3   $CRD_Print (CRD$GT_New_Line);
;    1225  3
;    1226  2      END;     ! End of INCR I FROM 0 TO 3 loop.
;    1227  2
;    1228  2      RETURN (CRD$K_Success);
;    1229  1 END;      ! End of MEN$Menu_Summary routine


          .EXTRN  DS$GETTERM, CRD$GT_INTERNAL_ERROR
          .EXTRN  CRD$GT_SUMMARY_TITLE
          .EXTRN  CRD$GT_ALL_FAILURES
          .EXTRN  CRD$GT_ALL_PASSES
          .EXTRN  CRD$GT_ALL_INC_HDWR_PREP
          .EXTRN  CRD$GT_NEW_LINE
          .EXTRN  CRD$GT_TAB, CRD$GT_SLASH

            0FFC 00000     .ENTRY  MEN$MENU_SUMMARY, Save R2,R3,R4,R5,R6,R7,-   ; 1003
           R8,R9,R10,R11            ;
          5E       1C  C2 00002     SUBL2   #28, SP          ;
    08    AE   7C 00005     CLRQ    SOME_PASS          ; 1030
    04    AE   D4 00008     CLRL    SOME_INC_PREP          ;
    14    AE   9F 0000B     PUSHAB  CONSOLE_CHAR          ; 1058
     00000000G  9F       01  FB 0000E    CALLS   #1, @#DS$GETTERM          ;
          0F       50  E9 00015     BLBC    R0, 1$          ;
    60    8F 15    AE  91 00018     CMPB    CONSOLE_CHAR+1, #96          ; 1067
```

```
                19  13 0001D    BEQL    2$              ;
              40   8F 15   AE  91 0001F    CMPB    CONSOLE_CHAR+1, #64          ; 1069
                12  13 00024    BEQL    2$              ;
                 04 00026    RET                 ; 1072
                00000000G  EF  9F 00027 1$:   PUSHAB  CRD$GT_INTERNAL_ERROR          ; 1075
                01  DD 0002D    PUSHL   #1              ;
                1A  DD 0002F    PUSHL   #26             ;
                00000000G  FF       03  FB 00031    CALLS   #3, @CRD$PRINT          ;
                00000000G  EF  9F 00038 2$:   PUSHAB  CRD$GT_SUMMARY_TITLE          ; 1081
                01  DD 0003E    PUSHL   #1              ;
                1A  DD 00040    PUSHL   #26             ;
                00000000G  FF       03  FB 00042    CALLS   #3, @CRD$PRINT          ;
                5A  D4 00049    CLRL    I               ; 1085
                01       5A  D1 0004B 3$:   CMPL    I, #1            ; 1092
                0C  12 0004E    BNEQ    4$              ;
                33 0C   AE  E9 00050    BLBC    SOME_FAIL, 7$           ;
                00000000G  EF  9F 00054    PUSHAB  CRD$GT_ALL_FAILURES         ;
                20  11 0005A    BRB     6$              ;
                02       5A  D1 0005C 4$:   CMPL    I, #2            ; 1093
                0C  12 0005F    BNEQ    5$              ;
                22 08   AE  E9 00061    BLBC    SOME_PASS, 7$           ;
                00000000G  EF  9F 00065    PUSHAB  CRD$GT_ALL_PASSES           ;
                0F  11 0006B    BRB     6$              ;
                03       5A  D1 0006D 5$:   CMPL    I, #3            ; 1094
                15  12 00070    BNEQ    7$              ;
                11 04   AE  E9 00072    BLBC    SOME_INC_PREP, 7$           ;
                00000000G  EF  9F 00076    PUSHAB  CRD$GT_ALL_INC_HDWR_PREP        ;
                01  DD 0007C 6$:   PUSHL   #1              ;
                1A  DD 0007E    PUSHL   #26             ;
             00000000G  FF       03  FB 00080    CALLS   #3, @CRD$PRINT          ;
                52  D4 00087 7$:   CLRL    TSTQ_TBL_INDEX          ; 1101
                53 00000000G  EF  7D 00089    MOVQ    MEN$GA_TEST_QUEUE, TSTQ_TBL_ENTRIES       ; 1102
                5B  D4 00090    CLRL    SCREEN_COUNTER          ; 1106
              00FB  31 00092    BRW     15$             ; 1114
50              14   A5       01      05  EF 00095 8$:   EXTZV   #5, #1, 20(DDB_PTR), R0        ; 1135
                57       50  90 0009B    MOVB    R0, FINISHED            ;
50              14   A5       01      04  EF 0009E    EXTZV   #4, #1, 20(DDB_PTR), R0        ; 1136
                51       50  90 000A4    MOVB    R0, BAD                 ;
58              14   A5       01      01  EF 000A7    EXTZV   #1, #1, 20(DDB_PTR), R8        ; 1137
                50       58  90 000AD    MOVB    R8, PREP_ERROR          ;
                6E       57  9A 000B0    MOVZBL  FINISHED, (SP)          ; 1142
                51       51  9A 000B3    MOVZBL  BAD, R1                 ;
                51         6E       51  CB 000B6    BICL3   R1, (SP), R1            ;
                57       50  9A 000BA    MOVZBL  PREP_ERROR, R7          ; 1144
                51       57  C8 000BD    BISL2   R7, R1                  ;
                58       51  92 000C0    MCOMB   R1, DDB_FAIL            ; 1142
57              50       01      00  EF 000C3    EXTZV   #0, #1, PREP_ERROR, R7        ; 1150
                51       57  90 000C8    MOVB    R7, DDB_INC_PREP        ;
                57       58  9A 000CB    MOVZBL  DDB_FAIL, R7            ; 1157
                6E       51  9A 000CE    MOVZBL  DDB_INC_PREP, (SP)          ;
                57       6E  C8 000D1    BISL2   (SP), R7                ;
                50       57  92 000D4    MCOMB   R7, DDB_PASS            ;
                57       58  9A 000D7    MOVZBL  DDB_FAIL, R7            ; 1159
             0C   AE       57  C8 000DA    BISL2   R7, SOME_FAIL           ;
                57       50  9A 000DE    MOVZBL  DDB_PASS, R7            ; 1160
```

```
      08   AE      57  C8 000E1    BISL2   R7, SOME_PASS          ;
           57      51  9A 000E5    MOVZBL  DDB_INC_PREP, R7         ; 1161
      04   AE      57  C8 000E8    BISL2   R7, SOME_INC_PREP        ;
           5A  D5 000EC    TSTL    I                ; 1165
           03  12 000EE    BNEQ    9$               ;
         009B  31 000F0    BRW     14$              ;
           01      5A  D1 000F3 9$:   CMPL    I, #1            ; 1166
           06  12 000F6    BNEQ    10$              ;
           16      58  E8 000F8    BLBS    DDB_FAIL, 12$          ;
         0090  31 000FB    BRW     14$              ;
           02      5A  D1 000FE 10$:  CMPL    I, #2            ; 1167
           06  12 00101    BNEQ    11$              ;
           0B      50  E8 00103    BLBS    DDB_PASS, 12$          ;
         0085  31 00106    BRW     14$              ;
           03      5A  D1 00109 11$:  CMPL    I, #3            ; 1168
           03  12 0010C    BNEQ    12$              ;
           7D      51  E9 0010E    BLBC    DDB_INC_PREP, 14$        ;
           56 08   A5  D0 00111 12$:  MOVL    8(DDB_PTR), PTABLE_PTR       ; 1177
           59 26   A6  9E 00115    MOVAB   38(R6), HDWR_NAME_PTR        ; 1179
           56  DD 00119    PUSHL   PTABLE_PTR          ; 1186
       00000000G  EF      01  FB 0011B    CALLS   #1, CRD$GET_DEVICE_NAME      ;
      10   AE      50  D0 00122    MOVL    R0, DEV_NAME           ;
           03      5B  D1 00126    CMPL    SCREEN_COUNTER, #3       ; 1194
           13  12 00129    BNEQ    13$              ;
       00000000G  EF  9F 0012B    PUSHAB  CRD$GT_NEW_LINE          ; 1197
           01  DD 00131    PUSHL   #1               ;
           1A  DD 00133    PUSHL   #26              ;
       00000000G  FF      03  FB 00135    CALLS   #3, @CRD$PRINT         ;
           5B  D4 0013C    CLRL    SCREEN_COUNTER          ; 1198
       00000000G  EF  9F 0013E 13$:   PUSHAB  CRD$GT_TAB           ; 1201
           01  DD 00144    PUSHL   #1               ;
           1A  DD 00146    PUSHL   #26              ;
       00000000G  FF      03  FB 00148    CALLS   #3, @CRD$PRINT         ;
           59  DD 0014F    PUSHL   HDWR_NAME_PTR           ; 1202
           01  DD 00151    PUSHL   #1               ;
           1A  DD 00153    PUSHL   #26              ;
       00000000G  FF      03  FB 00155    CALLS   #3, @CRD$PRINT         ;
       00000000G  EF  9F 0015C    PUSHAB  CRD$GT_SLASH           ; 1203
           01  DD 00162    PUSHL   #1               ;
           1A  DD 00164    PUSHL   #26              ;
       00000000G  FF      03  FB 00166    CALLS   #3, @CRD$PRINT         ;
      10   AE  DD 0016D    PUSHL   DEV_NAME            ; 1204
           01  DD 00170    PUSHL   #1               ;
           1A  DD 00172    PUSHL   #26              ;
       00000000G  FF      03  FB 00174    CALLS   #3, @CRD$PRINT         ;
       00000000G  EF  9F 0017B    PUSHAB  CRD$GT_TAB           ; 1205
           01  DD 00181    PUSHL   #1               ;
           1A  DD 00183    PUSHL   #26              ;
       00000000G  FF      03  FB 00185    CALLS   #3, @CRD$PRINT         ;
           5B  D6 0018C    INCL    SCREEN_COUNTER          ; 1207
           52  D6 0018E 14$:   INCL    TSTQ_TBL_INDEX          ; 1217
           53      52  D1 00190 15$:   CMPL    TSTQ_TBL_INDEX, TSTQ_TBL_ENTRIES       ; 1114
           09  18 00193    BGEQ    16$              ;
           55    6442  D0 00195    MOVL    @(TSTQ_TBL_PTR)[TSTQ_TBL_INDEX], DDB_PTR     ; 1115
           03  13 00199    BEQL    16$              ;
```

```
          FEF7  31 0019B     BRW     8$              ;
            03       5A  D1 0019E 16$:   CMPL    I, #3           ; 1222
            11  12 001A1     BNEQ    17$             ;
        00000000G  EF  9F 001A3     PUSHAB  CRD$GT_NEW_LINE         ; 1224
            01  DD 001A9     PUSHL   #1              ;
            1A  DD 001AB     PUSHL   #26             ;
        00000000G  FF       03  FB 001AD     CALLS   #3, @CRD$PRINT          ;
  FE91            5A        01      03  F1 001B4 17$:   ACBL    #3, #1, I, 3$           ; 1085
         04 001BA     RET                 ; 1229

; Routine Size:  443 bytes,    Routine Base:  CODE + 04B3
```

```
;    1230  1 END
;    1231  0 ELUDOM




;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          125  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK          256  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE         1646  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)




Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
$ASCIC     Macro    Lib02   164      403      429      894
$CRD_LITERALS    Macro    Lib03    97
$CRD_MESSAGE    Macro    Lib03   185      187      195      207      209      217      218      475      996
$CRD_PRINT    Unbound           185      187      195      207      209      217      218      475      996
    Macro    Lib03   164      185      187      195      207      209      217      218      339      429
              475      614      796      830      931      957      975      996     1075     1081
             1092     1093     1094     1197     1201     1202     1203     1204     1205     1224
$CRD_TERMCHAR_ATTR  Macro    Lib03  1050
$CRD_TERMCHAR_FLD  Fieldset Lib03  1050
$DS_ASKSTR    KeyWMacr Lib02   986
$DS_BREAK    Macro    Lib02   986
$DS_GETTERM    KeyWMacr Lib02  1058
$DS_HPO_DECL    Macro    Lib02   281      399      538      889     1038
$DS_HPO_F    Fieldset Lib02   281      399      538      889     1038
$DS_TYPEDEF    Macro    Lib02   164      185      187      195      207      209      217      218      344      430
              475      614      796      832      931      960      975      996     1075     1081
             1092     1093     1094     1197     1201     1202     1203     1204     1205     1224
$EQULST     Macro    Lib04    97       98       99      164      185      187      195      207      209      217
              218      344      430      475      614      796      832      931      960      975
              996     1075     1081     1092     1093     1094     1197     1201     1202     1203
             1204     1205     1224
$ER    Comptime    88
$MEN_ASKSTR    KeyWMacr Lib03   984
$MEN_LITERALS    Macro    Lib03    98
$MEN_SUPBLK_ATTR  Macro    Lib03   280      890     1037
$MEN_SUPBLK_FLD    Fieldset Lib03   280      890     1037
$MEN_TSTCNFG_ATTR  Macro    Lib03   891
$MEN_TSTCNFG_FLD  Fieldset Lib03   891
$MODULE    Bind         88
$SCREEN_LITERALS  Macro    Lib03    '9
AUTO$K_EXIT_AUTOSIZER_ERROR Literal      97
AUTO$K_EXIT_CONTROL_C  Literal     97
AUTO$K_EXIT_DUMMY_2  Literal     97
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
AUTO$K_EXIT_DUMMY_3  Literal     97
AUTO$K_EXIT_ERRORS_COMPLETE Literal     97
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     97
AUTO$K_EXIT_INTERNAL_ERROR Literal     97
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     97
AUTO$K_EXIT_LAST_REASON  Literal     97    97
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     97
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     97
AUTO$K_EXIT_NOERRORS_PREP Literal     97
BAD    Register  1124  1136=   1142.
BOOLEAN    Macro    Lib03  1151
CHAR_PTR   Register   395   409=    453=    453a=   456=    456a=   472.
COLON_PTR   Register   656   675=    679.    689.
COMMA_SPACE   Bind       403   453.
CONSOLE_CHAR   Local     1050  1058a   1067.   1069.
CRD$GB_CRD_DEBUG  External Lib03   163.    427.
CRD$GET_DEVICE_NAME  ExtRout  Lib03   325c    449c    788c    950c   1186c
CRD$GT_ALL_FAILURES  External Lib03  1092a
CRD$GT_ALL_INC_HDWR_PREP External Lib03  1094a
CRD$GT_ALL_PASSES  External Lib03  1093a
CRD$GT_INTERNAL_ERROR  External Lib03  1075a
CRD$GT_NEW_LINE   External Lib03  1197a   1224a
CRD$GT_NEW_LINE_MESSAGE  External Lib03   185a    187a    195a    207a    209a    217a    218a    475a    996a
CRD$GT_SLASH   External Lib03  1203a
CRD$GT_STAR_LINE_PREFIX_MESSAGE Unbound          185     187     195     207     209     217     218     475     996
CRD$GT_STAR_LINE_SUFFIX_MESSAGE Unbound          185     187     195     207     209     217     218     475     996
CRD$GT_SUMMARY_TITLE  External Lib03  1081a
CRD$GT_TAB   External Lib03  1201a   1205a
CRD$K_ACTION_EQL_DO_NOTHING Literal     97
CRD$K_ACTION_RANGE_ERROR Literal     97
CRD$K_ADVANCE_DIAGNOSTIC Literal     97
CRD$K_ADVANCE_GENERAL_COM Literal     97
CRD$K_ADVANCE_STATE  Literal     97
CRD$K_ALLOCATION_ERROR  Literal     97
CRD$K_ASSIGN_PTABLE_PTRS Literal     97
CRD$K_AUTOSIZER_ERROR  Literal     97
CRD$K_AUTOSIZER_SET_FLAGS Literal     97
CRD$K_BAD_ACTION_NUMBER  Literal     97
CRD$K_BAD_TYPECODE_ERROR Literal     97
CRD$K_BASE_SYSTEM_IDC  Literal     97
CRD$K_BASE_SYSTEM_LESI  Literal     97
CRD$K_BASE_SYSTEM_NULL  Literal     97
CRD$K_BASE_SYSTEM_UDA_0  Literal     97
CRD$K_BASE_SYSTEM_UDA_1  Literal     97
CRD$K_BASE_SYSTEM_UDA_2  Literal     97
CRD$K_BASE_SYSTEM_UDA_3  Literal     97
CRD$K_CRD_INITIALIZATION Literal     97
CRD$K_CREATE_HST  Literal     97
CRD$K_DATA_LENGTH_ERROR  Literal     97
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal     97
CRD$K_DDB_BLINK   Literal     97
CRD$K_DDB_FLINK   Literal     97
CRD$K_DDB_LAST_ENTRY  Literal     97
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_DDB_MEMORY_SIZE  Literal     97
CRD$K_DDB_MENU_DEVICE_NUMB Literal     97
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal     97   559
CRD$K_DDB_MENU_SUPPORT_SIZE Literal     97
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal     97   567
CRD$K_DDB_PTABLE_ENTRY  Literal     97   298     446     788     941    1177
CRD$K_DDB_STATUS  Literal     97   162     425    1135    1136    1137
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal     97   313     912
CRD$K_DEVICE_NAME  Literal     97
CRD$K_DIAGNOSTIC_ERROR  Literal     97
CRD$K_DIAGNOSTIC_NAME  Literal     97
CRD$K_DONE_GENERAL_COMS  Literal     97
CRD$K_DO_NOTHING  Literal     97
CRD$K_DUMMY_10   Literal     97
CRD$K_DUMMY_11   Literal     97
CRD$K_DUMMY_12   Literal     97
CRD$K_DUMMY_13   Literal     97
CRD$K_DUMMY_3   Literal     97
CRD$K_DUMMY_4   Literal     97
CRD$K_DUMMY_5   Literal     97
CRD$K_DUMMY_6   Literal     97
CRD$K_DUMMY_7   Literal     97
CRD$K_DUMMY_8   Literal     97
CRD$K_DUMMY_9   Literal     97
CRD$K_EXIT_CRD   Literal     97
CRD$K_HOOK_POINT  Literal     97
CRD$K_HS_INTERNAL_ERROR  Literal     97
CRD$K_IDC_EVDLB   Literal     97
CRD$K_IDC_EVDLC   Literal     97
CRD$K_IDC_EVDLD   Literal     97
CRD$K_IDC_EVRAD_0  Literal     97
CRD$K_IDC_EVRAD_1  Literal     97
CRD$K_IDC_LAST_DIAGNOSTIC Literal     97
CRD$K_INTERNAL_ERROR  Literal     97
CRD$K_LAST_ACTION_ROUTINE Literal     97
CRD$K_LAST_ENTRY  Literal     97   589
CRD$K_LAST_FUNCTION  Literal     97
CRD$K_LAST_HOOK_POINT  Literal     97
CRD$K_LAST_INTERNAL_ERROR Literal     97
CRD$K_LAST_TYPE   Literal     97
CRD$K_LESI_ENWCA  Literal     97
CRD$K_LESI_EVRMB_0  Literal     97
CRD$K_LESI_EVRMB_1  Literal     97
CRD$K_LESI_LAST_DIAGNOSTIC Literal     97
CRD$K_LEVEL_4_PASSED  Literal     97
CRD$K_LOAD_AUTOSIZER  Literal     97
CRD$K_LOAD_ERROR  Literal     97
CRD$K_LOAD_GENERAL_COM  Literal     97
CRD$K_LOAD_LOAD_COM  Literal     97
CRD$K_LOAD_SELECT_COM  Literal     97
CRD$K_LOAD_SET_EVENT_COM Literal     97
CRD$K_LOAD_SET_FLAGS_COM Literal     97
CRD$K_LOAD_START_COM  Literal     97
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_LOAD_SUP_FILE  Literal     97
CRD$K_MM_INTERNAL_ERROR  Literal      97
CRD$K_NEW_LINE   Literal      97
CRD$K_PREPARATION_ERROR  Literal      97
CRD$K_PREPARATION_ERROR_TEXT Literal      97
CRD$K_RUNTIME   Literal      97   332      579
CRD$K_SAME_LINE   Literal      97
CRD$K_SET_EVENT_ARGS  Literal      97
CRD$K_SET_FLAGS_ARGS  Literal      97
CRD$K_SET_UP_DIAGNOSTIC  Literal      97
CRD$K_START   Literal      97
CRD$K_START_AUTOSIZER  Literal      97
CRD$K_START_ERROR  Literal      97
CRD$K_START_QUALIFIERS  Literal      97
CRD$K_STAR_LINE   Literal      97
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal      97
CRD$K_STATE_ADVANCE_GENERAL_COM Literal      97
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal      97
CRD$K_STATE_AUTOSIZER_ERROR Literal      97
CRD$K_STATE_AUTOSIZER_RUNNING Literal      97
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal      97
CRD$K_STATE_DIAGNOSTIC_ERROR Literal      97
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal      97
CRD$K_STATE_DONE_DIAGNOSTICS Literal      97
CRD$K_STATE_DONE_GENERAL_COMS Literal      97
CRD$K_STATE_DUMMY_1  Literal      97
CRD$K_STATE_DUMMY_10  Literal      97
CRD$K_STATE_DUMMY_12  Literal      97
CRD$K_STATE_DUMMY_13  Literal      97
CRD$K_STATE_DUMMY_2  Literal      97
CRD$K_STATE_DUMMY_3  Literal      97
CRD$K_STATE_DUMMY_4  Literal      97
CRD$K_STATE_DUMMY_6  Literal      97
CRD$K_STATE_DUMMY_7  Literal      97
CRD$K_STATE_DUMMY_8  Literal      97
CRD$K_STATE_EXIT_CRD  Literal      97
CRD$K_STATE_INTERNAL_ERROR Literal      97
CRD$K_STATE_LAST_STATE  Literal      97
CRD$K_STATE_LEVEL_4_PASSED Literal      97
CRD$K_STATE_LOAD_AUTOSIZER Literal      97
CRD$K_STATE_LOAD_ERROR  Literal      97
CRD$K_STATE_LOAD_GENERAL_COM Literal      97
CRD$K_STATE_LOAD_LOAD_COM Literal      97
CRD$K_STATE_LOAD_SELECT_COM Literal      97
CRD$K_STATE_LOAD_SET_EVENT_COM Literal      97
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal      97
CRD$K_STATE_LOAD_START_COM Literal      97
CRD$K_STATE_PREPARATION_ERROR Literal      97
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal      97
CRD$K_STATE_START  Literal      97
CRD$K_STATE_START_AUTOSIZER Literal      97
CRD$K_STATE_START_ERROR  Literal      97
CRD$K_SUCCESS   Literal  Lib03   241      345      478      616      697      742      834      973      999     1228
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_TERMCHAR_SIZE  Literal  Lib03  1050
CRD$K_TEST_START_TEXT  Literal     97   308
CRD$K_TQ_INTERNAL_ERROR  Literal     97
CRD$K_TYPECODE   Literal     97
CRD$K_UDA_0_EVDLB  Literal     97
CRD$K_UDA_0_EVDLC  Literal     97
CRD$K_UDA_0_EVDLD  Literal     97
CRD$K_UDA_0_EVRLA_0  Literal     97
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     97
CRD$K_UDA_1_EVDLB  Literal     97
CRD$K_UDA_1_EVDLC  Literal     97
CRD$K_UDA_1_EVDLD  Literal     97
CRD$K_UDA_1_EVRLA_0  Literal     97
CRD$K_UDA_1_EVRLA_1  Literal     97
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     97
CRD$K_UDA_2_EVDLB  Literal     97
CRD$K_UDA_2_EVDLC  Literal     97
CRD$K_UDA_2_EVDLD  Literal     97
CRD$K_UDA_2_EVRLA_0  Literal     97
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     97
CRD$K_UDA_3_EVDLB  Literal     97
CRD$K_UDA_3_EVDLC  Literal     97
CRD$K_UDA_3_EVDLD  Literal     97
CRD$K_UDA_3_EVRLA_0  Literal     97
CRD$K_UDA_3_EVRLA_1  Literal     97
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     97
CRD$PRINT   External Lib03   164ac   185ac   187ac   195ac   207ac   209ac   217ac   218ac   344ac   430ac
             475ac   614ac   796ac   832ac   931ac   960ac   975ac   996ac  1075ac  1081ac
            1092ac  1093ac  1094ac  1197ac  1201ac  1202ac  1203ac  1204ac  1205ac  1224ac
CRD$SCAN$NUMERIC  External Lib03   733.
CRD$SCREEN_PAUSE  ExtRout  Lib03   238c
DDB    Bind       288   291a
DDB$V_STATUS_DEVICE_BAD  Macro    Lib03   167     433    1136
DDB$V_STATUS_DEV_TEST_COMPLETE Macro    Lib03   166    1135
DDB$V_STATUS_PREPARATION_ERROR Macro    Lib03  1137
DDB_FAIL   Register  1126  1142=   1157.   1159.   1166.
DDB_INC_PREP   Register  1128  1150=   1157.   1161.   1168.
DDB_PASS   Register  1127  1157=   1160.   1167.
DDB_PTR    Register   144   160=    162a.
    Register   278   291=    298a.   313a.
    Register   398   423=    425a.   430.    446a.
    Register   537   553=    559a.   567a.
    RoutForm   746   783m    788a.
    Register   888   910=    912a.   941a.
    Register  1036  1115=   1135a.  1136a.  1137a.  1177a.
DDB_STATUS   Register   145   162=    164.    166.    167.
    Register   400   425=    430.    433.
DEVICE_DATA_BLOCK_STRUCTURE Structur Lib03   144     278     398     537     783     888    1036
DEVICE_NAME   Local      437   449=    456a.
DEVICE_NAME_PTR   Register   786   788=    796.    832.
DEVNAM_STRING_BUF  Own         94   409a    472=    472a    475a
DEVNAM_STRING_BUF_SZ  Literal     92    94
DEV_NAME   Register   284   325=    344.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Register   886   950=    960.
    Local     1049  1186=   1204.
DIFFERENCE   Register   813
DS$ASKSTR   ExtRout    986   986c
DS$BREAK   ExtRout    986   986c    986e
DS$GETTERM   ExtRout   1058  1058c
DS$K_PRINTB   Literal    164
    Literal    185
    Literal    185
    Literal    187
    Literal    187
    Literal    195
    Literal    195
    Literal    207
    Literal    207
    Literal    209
    Literal    209
    Literal    217
    Literal    217
    Literal    218
    Literal    218
    Literal    344
    Literal    430
    Literal    475
    Literal    475
    Literal    614
    Literal    796
    Literal    832
    Literal    931
    Literal    960
    Literal    975
    Literal    996
    Literal    996
    Literal   1075
    Literal   1081
    Literal   1092
    Literal   1093
    Literal   1094
    Literal   1197
    Literal   1201
    Literal   1202
    Literal   1203
    Literal   1204
    Literal   1205
    Literal   1224
DS$K_PRINTF   Literal    164   164
    Literal    185   185
    Literal    185   185
    Literal    187   187
    Literal    187   187
    Literal    195   195
    Literal    195   195
    Literal    207   207
```

Symbol	Type	Defined	Referenced ...
	Literal	207	207
	Literal	209	209
	Literal	209	209
	Literal	217	217
	Literal	217	217
	Literal	218	218
	Literal	218	218
	Literal	344	344
	Literal	430	430
	Literal	475	475
	Literal	475	475
	Literal	614	614
	Literal	796	796
	Literal	832	832
	Literal	931	931
	Literal	960	960
	Literal	975	975
	Literal	996	996
	Literal	996	996
	Literal	1075	1075
	Literal	1081	1081
	Literal	1092	1092
	Literal	1093	1093
	Literal	1094	1094
	Literal	1197	1197
	Literal	1201	1201
	Literal	1202	1202
	Literal	1203	1203
	Literal	1204	1204
	Literal	1205	1205
	Literal	1224	1224
DS$K_PRINTI	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	
	Literal	832	

Symbol	Type	Defined	Referenced ...
	Literal	931	
	Literal	960	
	Literal	975	
	Literal	996	
	Literal	996	
	Literal	1075	
	Literal	1081	
	Literal	1092	
	Literal	1093	
	Literal	1094	
	Literal	1197	
	Literal	1201	
	Literal	1202	
	Literal	1203	
	Literal	1204	
	Literal	1205	
	Literal	1224	
DS$K_PRINTX	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	
	Literal	832	
	Literal	931	
	Literal	960	
	Literal	975	
	Literal	996	
	Literal	996	
	Literal	1075	
	Literal	1081	
	Literal	1092	
	Literal	1093	
	Literal	1094	
	Literal	1197	
	Literal	1201	
	Literal	1202	
	Literal	1203	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------ -   --- ---------------
    Literal   1204
    Literal   1205
    Literal   1224
DS$K_TYPE_ABORT_PROGRAM  Literal    164
    Literal    185
    Literal    185
    Literal    187
    Literal    187
    Literal    195
    Literal    195
    Literal    207
    Literal    207
    Literal    209
    Literal    209
    Literal    217
    Literal    217
    Literal    218
    Literal    218
    Literal    344
    Literal    430
    Literal    475
    Literal    475
    Literal    614
    Literal    796
    Literal    832
    Literal    931
    Literal    960
    Literal    975
    Literal    996
    Literal    996
    Literal   1075
    Literal   1081
    Literal   1092
    Literal   1093
    Literal   1094
    Literal   1197
    Literal   1201
    Literal   1202
    Literal   1203
    Literal   1204
    Literal   1205
    Literal   1224
DS$K_TYPE_ABORT_TEST  Literal    164
    Literal    185
    Literal    185
    Literal    187
    Literal    187
    Literal    195
    Literal    195
    Literal    207
    Literal    207
    Literal    209
    Literal    209
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------  -  -- ---------------
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_COMMAND_ERR  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------- ----- ---------------
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_COMMAND_OUT  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
```

Symbol	Type	Defined	Referenced ...
DS$K_TYPE_CRD_AUTOTEST	Literal	164	164
	Literal	185	185
	Literal	185	185
	Literal	187	187
	Literal	187	187
	Literal	195	195
	Literal	195	195
	Literal	207	207
	Literal	207	207
	Literal	209	209
	Literal	209	209
	Literal	217	217
	Literal	217	217
	Literal	218	218
	Literal	218	218
	Literal	344	344
	Literal	430	430
	Literal	475	475
	Literal	475	475
	Literal	614	614
	Literal	796	796
	Literal	832	832
	Literal	931	931
	Literal	960	960
	Literal	975	975
	Literal	996	996
	Literal	996	996
	Literal	996	996
	Literal	1075	1075
	Literal	1081	1081
	Literal	1092	1092
	Literal	1093	1093
	Literal	1094	1094
	Literal	1197	1197
	Literal	1201	1201
	Literal	1202	1202
	Literal	1203	1203
	Literal	1204	1204
	Literal	1205	1205
	Literal	1224	1224
DS$K_TYPE_DS_PROMPT	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ - - - -- ----- ---------------
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_DS_START  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ------ ---------

    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_ERRDEV  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_ERRHARD  Literal     164
    Literal     185
    Literal     185
```

Symbol	Type	Defined	Referenced ...
	Literal		187
	Literal		187
	Literal		195
	Literal		195
	Literal		207
	Literal		207
	Literal		209
	Literal		209
	Literal		217
	Literal		217
	Literal		218
	Literal		218
	Literal		344
	Literal		430
	Literal		475
	Literal		475
	Literal		614
	Literal		796
	Literal		832
	Literal		931
	Literal		960
	Literal		975
	Literal		996
	Literal		996
	Literal		1075
	Literal		1081
	Literal		1092
	Literal		1093
	Literal		1094
	Literal		1197
	Literal		1201
	Literal		1202
	Literal		1203
	Literal		1204
	Literal		1205
	Literal		1224
DS$K_TYPE_ERROR_BODY	Literal		164
	Literal		185
	Literal		185
	Literal		187
	Literal		187
	Literal		195
	Literal		195
	Literal		207
	Literal		207
	Literal		209
	Literal		209
	Literal		217
	Literal		217
	Literal		218
	Literal		218
	Literal		344
	Literal		430

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_ERROR_END  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_ERRPREP  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_ERRSOFT  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_ERRSUP  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_ERRSYS  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
```

Symbol	Type	Defined	Referenced ...
	Literal	1202	
	Literal	1203	
	Literal	1204	
	Literal	1205	
	Literal	1224	
DS$K_TYPE_ERR_HALT	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	
	Literal	832	
	Literal	931	
	Literal	960	
	Literal	975	
	Literal	996	
	Literal	996	
	Literal	1075	
	Literal	1081	
	Literal	1092	
	Literal	1093	
	Literal	1094	
	Literal	1197	
	Literal	1201	
	Literal	1202	
	Literal	1203	
	Literal	1204	
	Literal	1205	
	Literal	1224	
DS$K_TYPE_EXCEPTION	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    209
    Literal    209
    Literal    217
    Literal    217
    Literal    218
    Literal    218
    Literal    344
    Literal    430
    Literal    475
    Literal    475
    Literal    614
    Literal    796
    Literal    832
    Literal    931
    Literal    960
    Literal    975
    Literal    996
    Literal    996
    Literal   1075
    Literal   1081
    Literal   1092
    Literal   1093
    Literal   1094
    Literal   1197
    Literal   1201
    Literal   1202
    Literal   1203
    Literal   1204
    Literal   1205
    Literal   1224
DS$K_TYPE_EXCEPTION_HEAD Literal    164
    Literal    185
    Literal    185
    Literal    187
    Literal    187
    Literal    195
    Literal    195
    Literal    207
    Literal    207
    Literal    209
    Literal    209
    Literal    217
    Literal    217
    Literal    218
    Literal    218
    Literal    344
    Literal    430
    Literal    475
    Literal    475
    Literal    614
    Literal    796
    Literal    832
    Literal    931
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -- - ---   --- --------- ------
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_FIRST_PASS  Literal    164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
```

Symbol	Type	Defined	Referenced ...
	Literal	1205	
	Literal	1224	
DS$K_TYPE_GENERAL	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	
	Literal	832	
	Literal	931	
	Literal	960	
	Literal	975	
	Literal	996	
	Literal	996	
	Literal	1075	
	Literal	1081	
	Literal	1092	
	Literal	1093	
	Literal	1094	
	Literal	1197	
	Literal	1201	
	Literal	1202	
	Literal	1203	
	Literal	1204	
	Literal	1205	
	Literal	1224	
DS$K_TYPE_GENERAL_ERROR	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	

Symbol	Type	Defined	Referenced ...
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	
	Literal	832	
	Literal	931	
	Literal	960	
	Literal	975	
	Literal	996	
	Literal	996	
	Literal	1075	
	Literal	1081	
	Literal	1092	
	Literal	1093	
	Literal	1094	
	Literal	1197	
	Literal	1201	
	Literal	1202	
	Literal	1203	
	Literal	1204	
	Literal	1205	
	Literal	1224	
DS$K_TYPE_NO_TESTS	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	
	Literal	832	
	Literal	931	
	Literal	960	
	Literal	975	
	Literal	996	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ----- --  ---- ------- --------
    Literal    996
    Literal   1075
    Literal   1081
    Literal   1092
    Literal   1093
    Literal   1094
    Literal   1197
    Literal   1201
    Literal   1202
    Literal   1203
    Literal   1204
    Literal   1205
    Literal   1224
DS$K_TYPE_PARAM_ERROR  Literal    164
    Literal    185
    Literal    185
    Literal    187
    Literal    187
    Literal    195
    Literal    195
    Literal    207
    Literal    207
    Literal    209
    Literal    209
    Literal    217
    Literal    217
    Literal    217
    Literal    218
    Literal    218
    Literal    344
    Literal    430
    Literal    475
    Literal    475
    Literal    614
    Literal    796
    Literal    832
    Literal    931
    Literal    960
    Literal    975
    Literal    996
    Literal    996
    Literal   1075
    Literal   1081
    Literal   1092
    Literal   1093
    Literal   1094
    Literal   1197
    Literal   1201
    Literal   1202
    Literal   1203
    Literal   1204
    Literal   1205
    Literal   1224
DS$K_TYPE_PROGRAM_END  Literal    164
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_PROGRAM_INFO  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ - ------ ----- ---------------
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_PROGRAM_START  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----  - - ---- -------
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_QIO_INVADP  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_QIO_NODRIVER  Literal     164
    Literal     185
    Literal     185
    Literal     187
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_QIO_WRONGVER  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------  ------- ----- ---------------
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_SCRIPT_ECHO  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_SCRIPT_PNF  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_SCRIPT_PROMPT  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
```

Symbol	Type	Defined	Referenced ...
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	
	Literal	832	
	Literal	931	
	Literal	960	
	Literal	975	
	Literal	996	
	Literal	996	
	Literal	1075	
	Literal	1081	
	Literal	1092	
	Literal	1093	
	Literal	1094	
	Literal	1197	
	Literal	1201	
	Literal	1202	
	Literal	1203	
	Literal	1204	
	Literal	1205	
	Literal	1224	
DS$K_TYPE_SCRIPT_SKIP	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_SEQUENCE_ERROR Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_START_ERR  Literal    164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_START_LIST  Literal    164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
```

Symbol	Type	Defined	Referenced ...
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	
	Literal	832	
	Literal	931	
	Literal	960	
	Literal	975	
	Literal	996	
	Literal	996	
	Literal	1075	
	Literal	1081	
	Literal	1092	
	Literal	1093	
	Literal	1094	
	Literal	1197	
	Literal	1201	
	Literal	1202	
	Literal	1203	
	Literal	1204	
	Literal	1205	
	Literal	1224	
DS$K_TYPE_SUMMARY	Literal	164	
	Literal	185	
	Literal	185	
	Literal	187	
	Literal	187	
	Literal	195	
	Literal	195	
	Literal	207	
	Literal	207	
	Literal	209	
	Literal	209	
	Literal	217	
	Literal	217	
	Literal	218	
	Literal	218	
	Literal	344	
	Literal	430	
	Literal	475	
	Literal	475	
	Literal	614	
	Literal	796	
	Literal	832	
	Literal	931	
	Literal	960	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
    Literal    1224
DS$K_TYPE_USER_PROMPT  Literal     164
    Literal     185
    Literal     185
    Literal     187
    Literal     187
    Literal     195
    Literal     195
    Literal     207
    Literal     207
    Literal     209
    Literal     209
    Literal     217
    Literal     217
    Literal     218
    Literal     218
    Literal     344
    Literal     430
    Literal     475
    Literal     475
    Literal     614
    Literal     796
    Literal     832
    Literal     931
    Literal     960
    Literal     975
    Literal     996
    Literal     996
    Literal    1075
    Literal    1081
    Literal    1092
    Literal    1093
    Literal    1094
    Literal    1197
    Literal    1201
    Literal    1202
    Literal    1203
    Literal    1204
    Literal    1205
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal   1224
FAILED_HDWR_FOUND  Register   393   393=    459=    469.
FALSE    Literal  Lib03   140     190     199     212     222     393     883     884     981    1041
            1042    1043    1165
FINISHED   Register  1125  1135=   1142.
FNCTST_COMPLETE   Register   139   139=    166=    166.    175.
FNCTST_ERROR   Register   140   140=    167=    167.    178.    200.
GET1ST_    Macro    Lib04    97      98      99     164     185     187     195     207     209     217
             218     344     430     475     614     796     832     931     960     975
             996    1075    1081    1092    1093    1094    1197    1201    1202    1203
            1204    1205    1224
GET2ND_    Unbound            97      98      99     164     185     187     195     207     209     217
             218     344     430     475     614     796     832     931     960     975
             996    1075    1081    1092    1093    1094    1197    1201    1202    1203
            1204    1205    1224
HDWR_FOUND   Register   883   883=    933=    971.
HDWR_NAME_PTR   Register   283   300=    344.
    Register   885   943=    960.
    Local     1048  1179=   1202.
HP$K_LENGTH   Literal  Lib02   281     399     538     889    1038
HP$T_TYPE   Field    Lib02   300a    943a   1179a
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal     97
HS$K_ACTION_ADVANCE_GENERAL_COM Literal     97
HS$K_ACTION_ADVANCE_STATE Literal     97
HS$K_ACTION_CHANGE_TABLE Literal     97
HS$K_ACTION_DIAGNOSTIC_ERROR Literal     97
HS$K_ACTION_DONE_GENERAL_COMS Literal     97
HS$K_ACTION_DO_NOTHING  Literal     97
HS$K_ACTION_DUMMY_3  Literal     97
HS$K_ACTION_DUMMY_4  Literal     97
HS$K_ACTION_DUMMY_5  Literal     97
HS$K_ACTION_DUMMY_6  Literal     97
HS$K_ACTION_INIT_HS  Literal     97
HS$K_ACTION_INTERNAL_ERROR Literal     97
HS$K_ACTION_LAST_ACTION  Literal     97
HS$K_ACTION_LOAD_ERROR  Literal     97
HS$K_ACTION_LOAD_GENERAL_COM Litera      97
HS$K_ACTION_LOAD_LOAD_COM Literal     97
HS$K_ACTION_LOAD_SELECT_COM Literal     97
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     97
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     97
HS$K_ACTION_LOAD_START_COM Literal     97
HS$K_ACTION_PREPARATION_ERROR Literal     97
HS$K_ACTION_START_ERROR  Literal     97
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     97
HS$K_STATE_ADVANCE_GENERAL_COM Literal     97
HS$K_STATE_CHANGE_TABLE  Literal     97
HS$K_STATE_DIAGNOSTIC_ERROR Literal     97
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     97
HS$K_STATE_DONE_GENERAL_COMS Literal     97
HS$K_STATE_DUMMY_1  Literal     97
HS$K_STATE_DUMMY_2  Literal     97
HS$K_STATE_DUMMY_3  Literal     97
```

```
Symbol                          Type  Defined  Referenced ...
------  ----------------------- -------- ----- ---------------
HS$K_STATE_DJMMY_4  Literal     97
HS$K_STATE_DUMMY_6  Literal     97
HS$K_STATE_INIT_HS  Literal     97
HS$K_STATE_INTERNAL_ERROR Literal     97
HS$K_STATE_LAST_STATE  Literal     97
HS$K_STATE_LOAD_ERROR  Literal     97
HS$K_STATE_LOAD_GENERAL_COM Literal     97
HS$K_STATE_LOAD_LOAD_COM Literal     97
HS$K_STATE_LOAD_SELECT_COM Literal     97
HS$K_STATE_LOAD_SET_EVENT_COM Literal     97
HS$K_STATE_LCAD_SET_FLAGS_COM Literal     97
HS$K_STATE_LCAD_START_COM Literal     97
HS$K_STATE_PREPARATION_ERROR Literal     97
HS$K_STATE_START_ERROR  Literal     97
HST     Bind        289   292a
HST_INDEX   Register   535   569=    571=    571.
HST_NO    Register   534   567=    571.
HST_PTR    Register   279   292=    308a.   332a.
    Register   539   559=    579a.   589=    589.
I     Local     1085  1090.   1163.   1222.
I_DDB    RoutForm   245   288.
I_HST    RoutForm   245   289.
I_LENGTH   RoutForm   701   736.
I_MINUTES   RoutForm   620   664.
I_POINTER   RoutForm   701   737.
I_RETVAL   RoutForm   701   738.
I_SECONDS   RoutForm   620   665.
I_TIME    RoutForm   620   663.
JSB_SCAN_NUMERIC  Linkage  Lib03   733
LEFT    Register   809   822=    832.
LENGTH    Bind        736   740a
MEN$CVT_TIME_ASC_D  GlobRout   620 Lib03e     72f    582c
MEN$FNCTST_CMPLERR_STATUS GlobRout   102 Lib03e     68f
MEN$FNCTST_TESTSTRT_TEXT GlobRout   245 Lib03e     69f
MEN$GA_TEST_QUEUE  External Lib03   152.    153.    416.    417.    546.    547.    901.    902.   1102.   1103.
MEN$GB_TEST_IN_PROGRESS  External Lib03   222=
MEN$GET_TSTCLA_NAME  ExtRout  Lib03   315c
MEN$GK_TEST_QUEUE  Literal     98
MEN$GT_ASSISTANCE_MSG  External Lib03   218a
MEN$GT_CALL_FLDSRV  External Lib03   187a    209a
MEN$GT_FAILED_HDWR_NAME  External Lib03   475a
MEN$GT_FNCTST_CMPL_ERR  External Lib03   185a
MEN$GT_FNCTST_CMPL_NOERR External Lib03   195a
MEN$GT_FNCTST_INCMPL_ERR External Lib03   207a
MEN$GT_FNCTST_INCMPL_NOERR External Lib03   217a
MEN$GT_FNCTST_TESTSTRT_TEXT External Lib03   344a
MEN$GT_INVOPERINPUT  External Lib03   996a
MEN$GT_OPERINPUT_BUFFER  External Lib03   986a    993.
MEN$GT_PREP_ERROR_TEXT  External Lib03   806m    832a
MEN$GT_PREP_ERROR_TEXT_NO_USER External Lib03   796a
MEN$GT_RUNTIME_TOTAL  External Lib03   614a
MEN$GT_SCRMEDIA_HDWR  External Lib03   960a
MEN$GT_SCRMEDIA_HDWR_END External Lib03   975a
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- --------------
MEN$GT_SCRMEDIA_MSG  External Lib03   931a
MEN$GT_SCRMEDIA_PROMPT  External Lib03   986a
MEN$K_CNTLC_SEL_CONTINUE Literal     98
MEN$K_CNTLC_SEL_EXIT  Literal     98
MEN$K_CNTLC_SEL_FTMENU  Literal     98
MEN$K_CNTLC_SEL_LAST_ENTRY Literal     98
MEN$K_DEVTSTPREP_1  Literal     98
MEN$K_DEVTSTPREP_2  Literal     98
MEN$K_DEVTSTPREP_3  Literal     98
MEN$K_DEVTSTPREP_4  Literal     98
MEN$K_DEVTSTPREP_5  Literal     98
MEN$K_DEVTSTPREP_6  Literal     98
MEN$K_DEVTSTPREP_7  Literal     98
MEN$K_DEVTSTPREP_NULL  Literal     98
MEN$K_FTMRET_EXIT  Literal     98
MEN$K_FTMRET_FAILURE  Literal     98
MEN$K_FTMRET_MENU  Literal     98
MEN$K_FTMRET_TEST  Literal     98
MEN$K_FTMSEL_EXIT  Literal     98
MEN$K_FTMSEL_FNCTST_ALL  Literal     98
MEN$K_FTMSEL_LAST_ENTRY  Literal     98
MEN$K_FTMSEL_PREP  Literal     98
MEN$K_FTMSEL_SYSTEM_HDWR Literal     98
MEN$K_FTMSEL_TEST  Literal     98
MEN$K_HS_STATE_TABLE  Literal     97
MEN$K_MM_STATE_TABLE  Literal     97
MEN$K_PREP_ERROR_TEXT_LEN Literal     98
MEN$K_SELDEV_NUMB_START  Literal     98
MEN$K_SUPBLK_SIZE  Literal  Lib03   280     890    1037
MEN$K_TMENU_TSTALL_DEVNUMB Literal     98
MEN$K_TQ_STATE_TABLE  Literal     97
MEN$K_TSTCLA_ALL  Literal     98
MEN$K_TSTCLA_CARD  Literal     98
MEN$K_TSTCLA_COMM  Literal     98
MEN$K_TSTCLA_CPU  Literal     98
MEN$K_TSTCLA_DISK  Literal     98
MEN$K_TSTCLA_IO_ADAPTOR  Literal     98
MEN$K_TSTCLA_LAST_ENTRY  Literal     98
MEN$K_TSTCLA_MEMORY  Literal     98
MEN$K_TSTCLA_NULL  Literal     98
MEN$K_TSTCLA_PRINTER  Literal     98
MEN$K_TSTCLA_REAL  Literal     98
MEN$K_TSTCLA_TAPE  Literal     98
MEN$K_TSTCLA_TERM  Literal     98
MEN$K_TSTCNFG_SIZE  Literal  Lib03   891
MEN$K_TSTCTG_CPU  Literal     98
MEN$K_TSTCTG_IO_ADAPTOR  Literal     98
MEN$K_TSTCTG_IO_CONTROLLER Literal     98
MEN$K_TSTCTG_IO_UNIT  Literal     98
MEN$K_TSTCTG_MEMORY  Literal     98
MEN$K_TSTCTG_NULL  Literal     98
MEN$K_TSTTXT_DEVNAM_SZ  Literal     98
MEN$K_TSTTXT_TSTCLA_SZ  Literal     98
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ------------ --
MEN$K_TSTTXT_UUTNAM_SZ  Literal     98
MEN$MENU_SUMMARY  GlobRout  1003 Lib03e     75f    232c
MEN$PREPARATION_ERR_TEXT GlobRout   746 Lib03e     74f
MEN$PRINT_FAILED_HDWR  GlobRout   349 Lib03e     70f    186c    208c
MEN$PRINT_RUNTIME_TOTAL  GlobRout   482 Lib03e     71f
MEN$SCAN_NUMERIC_ASC_D  GlobRout   701 Lib03e     73f    682c    694c
MEN$SCRATCH_MEDIA_MSG  GlobRout   838 Lib03e     67f
MEN$V_TQ_TABLE_NUMB_ENTRIES Field    Lib03   152.    416.    546.    901.   1102.
MEN$V_TQ_TABLE_PTR  Field    Lib03   153.    417.    547.    902.   1103.
MENU$K_EXIT_AUTOSIZER_ERROR Literal     97
MENU$K_EXIT_INTERNAL_ERROR Literal     97
MENU$K_EXIT_LAST_REASON  Literal     97
MENU$K_EXIT_OPERATOR_ABORT Literal     97
MENU$K_EXIT_OPERATOR_STOP Literal     97
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     97
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     97
MINUTES    Bind       664   682a
MINUTES_PTR   Register   657   677=    679.    682.
MINUTES_SZ   Register   658   679=    682.
MINUTE_TOTAL   Register   529   529=    585=    585.    602=    602.    614.
MM$K_ACTION_ADVANCE_STATE Literal     97
MM$K_ACTION_AUTOSIZER_ERROR Literal     97
MM$K_ACTION_BUILD_DDBS  Literal     97
MM$K_ACTION_BUILD_HSTS  Literal     97
MM$K_ACTION_CHANGE_TABLE Literal     97
MM$K_ACTION_DONE_INITS  Literal     97
MM$K_ACTION_DO_NOTHING  Literal     97
MM$K_ACTION_DUMMY_3  Literal     97
MM$K_ACTION_DUMMY_4  Literal     97
MM$K_ACTION_DUMMY_5  Literal     97
MM$K_ACTION_DUMMY_6  Literal     97
MM$K_ACTION_DUMMY_7  Literal     97
MM$K_ACTION_DUMMY_8  Literal     97
MM$K_ACTION_INTERNAL_ERROR Literal     97
MM$K_ACTION_LAST_ACTION  Literal     97
MM$K_ACTION_LOAD_AUTOSIZER Literal     97
MM$K_ACTION_MENU_PROMPT  Literal     97
MM$K_ACTION_PRINT_MENU  Literal     97
MM$K_ACTION_PRINT_MENU_HEADING Literal     97
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     97
MM$K_ACTION_START_AUTOSIZER Literal     97
MM$K_STATE_AUTOSIZER_ERROR Literal     97
MM$K_STATE_AUTOSIZER_RUNNING Literal     97
MM$K_STATE_BUILD_DDBS  Literal     97
MM$K_STATE_BUILD_HSTS  Literal     97
MM$K_STATE_CHANGE_TABLE  Literal     97
MM$K_STATE_DONE_INITS  Literal     97
MM$K_STATE_DUMMY_1  Literal     97
MM$K_STATE_DUMMY_2  Literal     97
MM$K_STATE_DUMMY_3  Literal     97
MM$K_STATE_DUMMY_5  Literal     97
MM$K_STATE_DUMMY_7  Literal     97
MM$K_STATE_DUMMY_8  Literal     97
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_STATE_INTERNAL_ERROR Literal     97
MM$K_STATE_LAST_STATE  Literal     97
MM$K_STATE_LOAD_AUTOSIZER Literal     97
MM$K_STATE_MENU_PROMPT  Literal     97
MM$K_STATE_PRINT_MENU  Literal     97
MM$K_STATE_PRINT_MENU_HEADING Literal     97
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     97
MM$K_STATE_START  Literal     97
MM$K_STATE_START_AUTOSIZER Literal     97
MODNAM    Macro    Lib01    88
NAME_PTR   Register   277   308=    310.    310a.   319.
NUL2ND_    Macro    Lib04    97      98      99     164     185     187     195     207     209     217
             218     344     430     475     614     796     832     931     960     975
             996    1075    1081    1092    1093    1094    1197    1201    1202    1203
            1204    1205    1224
NUMB_BAD_DEVICES  Register   396   396=    439=    439.    452.
POINTER    Bind       737   740a
PREP_ERROR   Register  1123  1137=   1144.   1151.
PREP_TEXT_LEN   Register   811
PRINTED_BASIC_MSG  Register   884   884=    928.    932=
PTABLE_PTR   Register   281   298=    300aa   325.
    Register   399   446=    449.
    Register   538
    Register   889   941=    943aa   950.
    Register  1038  1177=   1179aa  1186.
RETVAL    Bind       738   740=
RIGHT    Register   810   823=    832.
RUNTIME_MINUTES   Local      524   577=    582a    585.
RUNTIME_PTR   Register   285   332=    344.
    Register   536   579     582.
RUNTIME_SECONDS   Local      525   578=    582a    587.
SCAN_NUMERIC   BindRout   733   740c
SCREEN$K_CLEAR_SCREEN  Literal     99
SCREEN$K_COUNT_LINE  Literal     99
SCREEN$K_PRESS_RETURN  Literal     99
SCREEN$K_PRESS_RETURN_MM Literal     99   238
SCREEN$K_PRESS_RETURN_TM Literal     99
SCREEN$K_TM_MSG   Literal     99
SCREEN_COUNTER   Local     1051  1106=   1194.   1198=   1207=   1207.
SECONDS    Bind       665   694a
SECONDS_PTR   Register   659   689=    691.    694.
SECONDS_SZ   Register   660   691=    694.
SECOND_REMAINING  Register   528   608=    614.
SECOND_TOTAL   Register   530   530=    587=    587.    602.    608.
SOME_FAIL   Local     1041  1041=   1092.   1159=   1159.
SOME_INC_PREP   Local     1043  1043=   1094.   1161=   1161.
SOME_PASS   Local     1042  1042=   1093.   1160=   1160.
STATUS    Local      986   986=    986.
SUPBLK$B_STRTFRST_CNFGBLK Field    Lib03   914a
SUPBLK$B_TEST_CLASS  Field    Lib03   315a
SUPBLK_PTR   Register   280   313=    315aa
    Register   890   912=    914aa
    Register  1037
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------- -------- ----- ---------------
TERMCHAR$B_TYPE   Field    Lib03  1067.   1069.
TIME    Bind        663   667a
TIME_END_PTR   Register   654   673=    675.    691.
TIME_PTR   Register   655   667=    673a.   673aa   675aa   675a.   677aa
TQ$K_ACTION_ADVANCE_STATE Literal     97
TQ$K_ACTION_CHANGE_TABLE Literal     97
TQ$K_ACTION_DONE_TEST_QUEUE Literal     97
TQ$K_ACTION_DO_NOTHING  Literal     97
TQ$K_ACTION_DUMMY_3  Literal     97
TQ$K_ACTION_DUMMY_4  Literal     97
TQ$K_ACTION_DUMMY_5  Literal     97
TQ$K_ACTION_GET_DDB  Literal     97
TQ$K_ACTION_INIT_TQ  Literal     97
TQ$K_ACTION_INTERNAL_ERROR Literal     97
TQ$K_ACTION_LAST_ACTION  Literal     97
TQ$K_STATE_CHANGE_TABLE  Literal     97
TQ$K_STATE_DONE_TEST_QUEUE Literal     97
TQ$K_STATE_DUMMY_1  Literal     97
TQ$K_STATE_DUMMY_2  Literal     97
TQ$K_STATE_DUMMY_3  Literal     97
TQ$K_STATE_DUMMY_4  Literal     97
TQ$K_STATE_DUMMY_5  Literal     97
TQ$K_STATE_GET_DDB  Literal     97
TQ$K_STATE_INIT_TQ  Literal     97
TQ$K_STATE_INTERNAL_ERROR Literal     97
TQ$K_STATE_LAST_STATE  Literal     97
TRUE    Literal  Lib03   139     177     180     202     459     932     933
TSTCLA_NAME_PTR   Local      274   315a    319=    344.
TSTCNFG$V_STATUS_SCRATCH Field    Lib03   921.
TSTCNFG_PTR   Register   891   914=    921a.
TSTQ_TBL_ENTRIES  Register   142   152=    160.
    Local       389   416=    423.
    Register    532   546=    553.
    Local       878   ,01=    910.
    Register   1034  1102=   1114.
TSTQ_TBL_INDEX   Register   141   151=    160.    168=    168.
    Local       388   415=    423.    462=    462.
    Register    531   545=    553.    594=    594.
    Local       877   900=    910.    963=    963.
    Register   1033  1101=   1114.   1115.   1217=   1217.
TSTQ_TBL_PTR   Register   143   153=    160a.   230a.
    Local       390   417=    423a.
    Register    533   547=    553a.
    Local       879   902=    910a.
    Register   1035  1103=   1115a.
TT$_VT100   Literal  Lib04  1067
TT$_VTS2   Literal  Lib04  1069
T_PROMPT_DEFAULT  Bind        894   986a    994.
USER_TEXT_LEN   Register   812
USER_TEXT_PTR   RoutForm   746   782m    795.    832.
VALID_STATUS   Register   882   981=    993=    997.
```

```
  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]MENTST.B32;1
    3  Module MENTST
   60  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   61  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   62  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   63  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
 1231  Eludom MENTST


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     @ Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          1          0          46           00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923         21          2          94           00:00.1
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486         90         18          62           00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          5           0       1000           00:04.2




;     COMMAND QUALIFIERS

; BLISS MENTST.B32/LIST=MENTST.LIS/CROSS_REFERENCE/DEBUG/OBJECT=MENTST.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  1646 code + 381 data bytes
; Run Time:    01:36.3
; Elapsed Time:    01:51.1
; Lines/CPU Min:     766
; Lexemes/CPU-Min: 76981
```

```
; Memory Used: 471 pages
; Compilation Complete
```

```
;    0001   0 %TITLE  'CRD MENU - Test Initialization'
;    0002   0 MODULE MENTSTINI  (
;    0003   0   IDENT = 'V01.4'
;    0004   0   ) =
;    0005   0 !
;    0006   0 ! COPYRIGHT (C)  1982,1985
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 !
;    0024   0 !++
;    0025   0 ! FACILITY:
;    0026   0 !
;    0027   0 ! ABSTRACT:
;    0028   0 !
;    0029   0 ! ENVIRONMENT:
;    0030   0 !
;    0031   0 ! AUTHOR:
;    0032   0 !
;    0033   0 ! John Ciukaj  May-1982
;    0034   0 !
;    0035   0 ! CREATION DATE:
;    0036   0 !
;    0037   0 ! VERSION:
;    0038   0 !
;    0039   0 ! Modified by:
;    0040   0 !
;    0041   0 ! 01 John Ciukaj Version 1.2 4-April-1983
;    0042   0 !  - changed references to CRD bits of DSA Flags to
;    0043   0 !    distinguish between Offline and Online.
;    0044   0 !
;    0045   0 ! 02 rwp 3-OCT-1984
;    0046   0 !  Added code to handle CRD online
;    0047   0 !
;    0048   0 ! 03 Ronald W. Parker 28-JAN-1985
;    0049   0 !  - Added code to  test for ONLINE vs. OFFLINE diagnostics
;    0050   0 !    that are in the EVLAA data base.
;    0051   0 !
;    0052   0 ! 04 Amy Iorio  15-AUG-1985
;    0053   0 !  - Substituted RMS for $DS_LOAD call.
;    0054   0 !
;    0055   0 ! 05      Ronald W. Parker
```

```
;    0056  0 !  - Changed EVLAA back to ENLAA
;    0057  0 !--
;    0058  0 %SBTTL 'Libraries'
;    0059  1 BEGIN
;    0060  1
;    0061  1  LIBRARY
;    0062  1    '$DS';
;    0063  1  LIBRARY
;    0064  1    '$DIAG';
;    0065  1  LIBRARY
;    0066  1    '$CRD';
;    0067  1  LIBRARY
;    0068  1    'SYS$LIBRARY:LIB';
;    0069  1
;    0070  1  BUILTIN
;    0071  1    INSQUE,
;    0072  1    REMQUE;
;    0073  1 %SBTTL 'Forward-External  Routines'
;    0074  1
;    0075  1 FORWARD ROUTINE
;    0076  1
;    0077  1  MEN$Menu_FncTst_Init  : ADDRESSING_MODE (LONG_RELATIVE),
;    0078  1  MEN$Build_DDB    : ADDRESSING_MODE (LONG_RELATIVE),
;    0079  1  MEN$Determine_Support  : ADDRESSING_MODE (LONG_RELATIVE),
;    0080  1  Create_HST    : ADDRESSING_MODE (LONG_RELATIVE),
;    0081  1  MEN$Specify_SelectNo  : ADDRESSING_MODE (LONG_RELATIVE),
;    0082  1  MEN$Allocate_Memory  : ADDRESSING_MODE (LONG_RELATIVE),
;    0083  1  MEN$Load_File    : ADDRESSING_MODE (LONG_RELATIVE),
;    0084  1  MEN$Find_SupBlk  : ADDRESSING_MODE (LONG_RELATIVE),
;    0085  1  MEN$Dump_TstQ_HST : ADDRESSING_MODE (LONG_RELATIVE),
;    0086  1  MEN$Menu_Cleanup  : ADDRESSING_MODE (LONG_RELATIVE);
;    0087  1 %SBTTL 'Psect Definitions'
;    0088  1
;    0089  1 PSECT
;    0090  1     PLIT = Data (NOEXECUTE, SHARE,   NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0091  1 PSECT
;    0092  1     CODE = Code (EXECUTE,   SHARE,   NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0093  1 PSECT
;    0094  1     OWN  = Work (NOEXECUTE, NOSHARE, WRITE,   ADDRESSING_MODE (LONG_RELATIVE));
;    0095  1 PSECT
;    0096  1     GLOBAL = Work (NOEXECUTE, NOSHARE, WRITE,   ADDRESSING_MODE (LONG_RELATIVE));
;    0097  1 %SBTTL 'Module-Global Static Storage'
;    0098  1
;    0099  1      Modnam ('MENTSTINI');   ! Module Name for Errsup Macro
;    0100  1 %SBTTL 'Macro - Literal Definitions'
;    0101  1
;    0102  1  $CRD_Literals;
;    0103  1  $MEN_Literals;
;    0104  1  $DS_DSADef;   ! Define the DSA flags
;    0105  1 %SBTTL 'Macro Definitions'
;    0106  1
;    0107  1 MACRO
; M  0108  1  Internal_Error_Exit (Type, Size_1, Size_2, Addr) =
; M  0109  1   BEGIN
; M  0110  1   MEN$Menu_Test_Internal_Error (Type, Size_1, Size_2, Addr);
```

```
;  M 0111  1    CRD$Stop (MENU$K_Exit_Internal_Error);
;  M 0112  1    END
;    0113  1   %;
;    0114  1
;    0115  1 MACRO
;  M 0116  1  Load_Error_Exit (RMS_Status) =
;  M 0117  1    BEGIN
;  M 0118  1    IF (RMS_Status EQL RMS$_FNF) THEN
;  M 0119  1     CRD$Stop (MENU$K_Exit_Sup_File_Not_Found)
;  M 0120  1    ELSE
;  M 0121  1     CRD$Stop (MENU$K_Exit_Sup_File_Load_Err)
;  M 0122  1    END
;    0123  1   %;
;    0124  1 %SBTTL 'Module Storage'
;    0125  1
;    0126  1 GLOBAL
;    0127  1
;    0128  1      MEN$GA_DDB_Head    : $MEN_Queue_Link_ATTR
;    0129  1        PRESET (
;    0130  1       [Queue$L_FLink] = 0,
;    0131  1       [Queue$L_BLink] = 0
;    0132  1        ),
;    0133  1
;    0134  1      MEN$GA_FuncSup_DatBase : $MEN_FuncSup_DatBase_ATTR
;    0135  1        PRESET (
;    0136  1       [MEN$V_FuncSup_DatBase_Size]    = 0,
;    0137  1       [MEN$V_FuncSup_DatBase_Ptr]     = 0,
;    0138  1       [MEN$V_FuncSup_DatBase_Total_Siz] = 0,
;    0139  1       [MEN$V_FuncSup_DatBase_Start] = 0
;    0140  1        ),
;    0141  1
;    0142  1      ! longword array pointing to ASCII Support
;    0143  1      ! database for CRD MENU Functional Testing
;    0144  1      !   [0] = Size in bytes of database
;    0145  1      !   [1] = Address pointer to first device
;    0146  1      !         support block
;    0147  1      !   [2] = Size in bytes of allocated memory
;    0148  1      !   for database
;    0149  1      !   [3] = Address pointer to start of
;    0150  1      !         loaded database
;    0151  1
;    0152  1      MEN$GA_Test_Queue   : $MEN_Test_Queue_ATTR
;    0153  1        PRESET (
;    0154  1       [MEN$V_TQ_Table_Numb_Entries] = 0,
;    0155  1       [MEN$V_TQ_Table_Ptr]        = 0,
;    0156  1       [MEN$V_TQ_Table_Size]        = 0
;    0157  1        ),
;    0158  1
;    0159  1      ! longword array pointing to Test Queue
;    0160  1      ! DDB pointer Table for CRD MENU Functional Testing
;    0161  1      !   [0] = Number of longword entries in Table
;    0162  1      !   [1] = Address pointer to table and first
;    0163  1      !    DDB pointer entry.  DDB pointers
;    0164  1      !    are valid until first null DDB pointer entry.
;    0165  1      !   [2] = Size in bytes of allocated memory
```

```
;    0166  1      !          space for Table.
;    0167  1
;    0168  1      MEN$GB_FncTst_Init  : BYTE INITIAL (BYTE (False)),
;    0169  1
;    0170  1      ! Flag location indicating Functional Test
;    0171  1      ! Initialization Status
;    0172  1      ! 0 or False = Init not complete
;    0173  1      ! 1 or True = Init complete
;    0174  1
;    0175  1      MEN$GB_Test_In_Progress : BYTE INITIAL (BYTE (False));
;    0176  1
;    0177  1      ! Flag location indicating Diagnostic Test Status
;    0178  1      ! 0 or False = Diagnostic testing is not in progress
;    0179  1      ! 1 or True = Diagnostic testing is in progress
;    0180  1  %SBTTL 'Support File Names'
;    0181  1
;    0182  1  OWN
;    0183  1      A_MenFuncSup_File : BLOCK [DSC$K_S_BLN, BYTE],
;    0184  1        ! Descriptor for Functional Test Support File
;    0185  1      A_File_Default    : BLOCK [DSC$K_S_BLN, BYTE];
;    0186  1        ! Descriptor for null file
;    0187  1  GLOBAL BIND
;    0188  1      MEN$GT_MenFuncSup_File = $ASCIC ('ENLAA.SUP')  : VECTOR [,BYTE],
;    0189  1      MEN$GT_File_Default    = $ASCIC ('')           : VECTOR [,BYTE];
;    0190  1  %SBTTL 'CRD MENU Functional Test Initialization Routine  '
;    0191  1
;    0192  1  GLOBAL ROUTINE MEN$Menu_FncTst_Init (DSA_Extract)=
;    0193  1
;    0194  1  !++
;    0195  1  ! FUNCTIONAL DESCRIPTION:
;    0196  1  !
;    0197  1  !  Initialization for CRD MENU TEST Functional Testing.  Should be called
;    0198  1  !  when CRD MENU TEST is invoked.
;    0199  1  !
;    0200  1  !     . Initializes CRD MENU TEST Flags
;    0201  1  !     . Initializes MENU Tables
;    0202  1  !     . Creates DDB database
;    0203  1  !     . Allocates Memory for Test Queue Tables
;    0204  1  !     . Determines CRD MENU TEST Hardware Support
;    0205  1  !     . Creates Hardware Support Tables for supported hardware
;    0206  1  !
;    0207  1  ! Input:
;    0208  1  !    DSA_Extract - this is a longword value passed from CRDINTRFC
;    0209  1  !
;    0210  1  ! OUTPUT PARAMETERS
;    0211  1  !
;    0212  1  ! IMPLICIT INPUTS
;    0213  1  !
;    0214  1  !    It is expected that VDS P-Tables exist prior to this
;    0215  1  !    routine call representing system hardware.
;    0216  1  !
;    0217  1  !    The Functional Test ASCII Support File resides on load media.
;    0218  1  !
;    0219  1  ! SIDE EFFECTS
;    0220  1  !
```

```
;    0221  1 !    Fatal Failures during this routine may cause CRD MENU TEST
;    0222  1 !    to abort by calling CRD$STOP.  Therefore, a return may never
;    0223  1 !    be made to the caller.
;    0224  1 !
;    0225  1 ! COMPLETION CODES
;    0226  1 !
;    0227  1 ! CRD$K_Success   Success
;    0228  1 ! CRD$K_Failure  Failure
;    0229  1 ! History:
;    0230  1 !
;    0231  1 !   01  ---  ?-???-??  Created routine
;    0232  1 !   02  rwp  3-Oct-84  Modified to handle CRD online condition
;    0233  1 !--
```

```
;    0234  1
;    0235  2 BEGIN
;    0236  2     IF (.CRD$GB_CRD_Debug) THEN
;    0237  2  $CRD_Print ($ASCIC ('MEN$FncTst_Init: Starting the Menu Test Initialization!/'));
;    0238  2
;    0239  2     CODECOMMENT
;    0240  2  '',
;    0241  2  ' Initialize flags',
;    0242  2  '':
;    0243  3  BEGIN
;    0244  3  MEN$GB_FncTst_Init = False; ! Indicate that Initialization has not been completed
;    0245  2  END;
;    0246  2
;    0247  2     CODECOMMENT
;    0248  2  '',
;    0249  2  'Initialize things common to both Auto Test and Menu Test',
;    0250  2  '':
;    0251  3  BEGIN
;    0252  3  CRD$Common_Initializations (MEN$CntIC_Menu);
;    0253  3      ! Pass the address of the Menu Test's ^C handler routine.
;    0254  3  DSA$V_Binary       = On; ! Turn these two on after the initializations are done.
;    0255  3      ! Leave them on for the duration of CRD Menu Test.  [01]
;    0256  3  SELECTONE .DSA_Extract OF
;    0257  3      SET
;    0258  3      [ 2 ] : DSA$V_CRD_MenuTest_Off = On;
;    0259  3      [ 4 ] : DSA$V_CRD_Menutest_On  = On;
;    0260  3      [ OTHERWISE ] : CRD$Stop (MENU$K_Exit_Internal_Error);
;    0261  3      TES;
;    0262  2  END;
;    0263  2
;    0264  2     CODECOMMENT
;    0265  2  '',
;    0266  2  'Initialize Menu tables.',
;    0267  2  'Since the  CRD MENU TEST  control code is a loadable image,',
;    0268  2  'link-time  resolved  address  pointers  must have an offset',
;    0269  2  'added to them.  This  is  because  the address pointers are',
;    0270  2  'resolved at link time with reference to a zero base address',
;    0271  2  'However,  at  run  time,  the  code  will  be  loaded at an',
;    0272  2  'address starting at CRD$AL_CRD_Dispatch_Vectors. Therefore,',
;    0273  2  'for  all  tables with text address pointers,  add an offset',
;    0274  2  'equal to the address of CRD$AL_CRD_Dispatch_Vectors.':
;    0275  3  BEGIN
;    0276  3  !+
;    0277  3  ! Initialize Address Pointers for the Functional Test Menu Table
;    0278  3  !-
;    0279  3
;    0280  3  MEN$FTMTbl_Init ();
;    0281  3
;    0282  3  !+
;    0283  3  ! Initialize Address Pointers for the Device Test Preparation Table
;    0284  3  !
;    0285  3
;    0286  3  MEN$PrepTbl Init ();
;    0287  3
;    0288  3  !+
```

```
;    0289  3  ! Initialize Address Pointers for the Control C Initialization Menu Table
;    0290  3  !-
;    0291  3
;    0292  3  MEN$CntlCInit_Tbl_Init ();
;    0293  3
;    0294  3  !+
;    0295  3  ! Initialize Address Pointers for the Control C Menu Table
;    0296  3  !-
;    0297  3
;    0298  3  MEN$Cntlc_Tbl_Init ();
;    0299  2  END;
;    0300  2
;    0301  2     CODECOMMENT
;    0302  2  '',
;    0303  2  'Initialize the General Commands Table',
;    0304  2  'and the Menu State tables:',
;    0305  2  '':
;    0306  3  BEGIN
;    0307  3  CRD$GB_Current_State_Table = MEN$K_MM_State_Table;
;    0308  3      ! Start out with the Main Menu State Table
;    0309  3  CRD$Init_General_Com_Table (); ! Initialize the General Command Table
;    0310  3  MEN$Init_MM_State_Table (); ! Init the Main Menu State Table
;    0311  3  MEN$Init_TQ_State_Table (); ! Init the Test Queue State Table
;    0312  3  MEN$Init_HS_State_Table (); ! Init the Hardware Support State Table
;    0313  2  END;
;    0314  2
;    0315  2     CODECOMMENT
;    0316  2  '',
;    0317  2  'Initialization complete',
;    0318  2  '':
;    0319  3  BEGIN
;    0320  3  DSA$V_Binary       = On; ! Turn these two on after the initializations are done.
;    0321  3  SELECTONE .DSA_Extract OF
;    0322  3      SET
;    0323  3      [ 2 ] : DSA$V_CRD_MenuTest_Off = On;
;    0324  3      [ 4 ] : DSA$V_CRD_Menutest_On  = On;
;    0325  3      [ OTHERWISE ] : CRD$Stop (MENU$K_Exit_Internal_Error);
;    0326  3      TES;
;    0327  2  END;
;    0328  2
;    0329  2     IF (.CRD$GB_CRD_Debug) THEN
;    0330  2  $CRD_Print ($ASCIC ('MEN$FncTst_Init: All done the Menu Test Initialization!/'));
;    0331  2
;    0332  2     RETURN (CRD$K_Success);  ! The initialization succeeded
;    0333  1 END;      ! Routine MEN$Menu_FncTst_Init


        .TITLE  MENTSTINI CRD MENU - Test Initialization
        .IDENT  \V01.4\

        .PSECT  WORK,NOEXE,2

 00000000  00000000  00000 MEN$GA_DDB_HEAD::
        .LONG   0, 0            ;
  00000000  00000000  00000000  00000000  00008 MEN$GA_FUNCSUP_DATBASE::
```

```
                .LONG   0, 0, 0, 0              ;
              00000000  00000000  00000000  00018 MEN$GA_TEST_QUEUE::
                .LONG   0, 0, 0                 ;
            00  00024 MEN$GB_FNCTST_INIT::
                .BYTE   0                       ;
            00  00025 MEN$GB_TEST_IN_PROGRESS::
                .BYTE   0                       ;
                00026   .BLKB   2
                00028 A_MENFUNCSUP_FILE:
                .BLKB   8
                00030 A_FILE_DEFAULT:
                .BLKB   8

                .PSECT  DATA,NOWRT,NOEXE,  SHR,2

           49  4E  49  54  53  54  4E  45  4D  09  00000 P.AAA:  .ASCII  <9>\MENTSTINI\          ;
           50  55  53  2E  41  41  4C  4E  45  09  0000A P.AAB:  .ASCII  <9>\ENLAA.SUP\          ;
             00  00014 P.AAC:  .ASCII  <0>                  ;
69  6E  49  5F  74  73  54  63  6E  46  24  4E  45  4D  38  00015 P.AAD:  .ASCII  \8MEN$FncTst_Init: Starting the Menu Test\  ;
65  68  74  20  67  6E  69  74  72  61  74  53  20  3A  74  00024                 ;
          74  73  65  54  20  75  6E  65  4D  20  00033                 ;
6E  6F  69  74  61  7A  69  6C  61  69  74  69  6E  49  20  0003D    .ASCII  \ Initialization!/\          ;
              2F  21  0004C                 ;
69  6E  49  5F  74  73  54  63  6E  46  24  4E  45  4D  38  0004E P.AAE:  .ASCII  \8MEN$FncTst_Init: All done the Menu Test\  ;
65  68  74  20  65  6E  6F  64  20  6C  6C  41  20  3A  74  0005D                 ;
          74  73  65  54  20  75  6E  65  4D  20  0006C                 ;
6E  6F  69  74  61  7A  69  6C  61  69  74  69  6E  49  20  00076    .ASCII  \ Initialization!/\          ;
              2F  21  00085                 ;

$MODULE=        P.AAA
DSA$AL_APTMAIL=      65024
DSA$GL_FLAGS=        65024
DSA$GL_APTCOM=       65028
DSA$GL_PASSES=       65032
DSA$GL_UNITS=        65036
DSA$GL_SECTNO=       65040
DSA$GL_SID=         65044
DSA$GL_MSGTYP=       65088
DSA$GL_ERRNO=        65092
DSA$GL_EVENT=        65096
DSA$GL_SUBTNO=       65100
DSA$GL_TESTNO=       65104
DSA$GL_PASSNO=       65108
DSA$GL_DEVLEN=       65112
DSA$GL_DEVNAM=       65116
DSA$GL_MSGPTR=       65128
MEN$GT_MENFUNCSUP_FILE==
        P.AAB
MEN$GT_FILE_DEFAULT==
        P.AAC
  .EXTRN  MEN$MENU_FNCTST_INIT
  .EXTRN  MEN$BUILD_DDB, MEN$DETERMINE_SUPPORT
  .EXTRN  MEN$SPECIFY_SELECTNO
  .EXTRN  MEN$ALLOCATE_MEMORY
  .EXTRN  MEN$LOAD_FILE, MEN$FIND_SUPBLK
```

```
                .EXTRN  MEN$DUMP_TSTQ_HST
                .EXTRN  MEN$MENU_CLEANUP
                .EXTRN  MEN$MENU_TEST_INTERNAL_ERROR
                .EXTRN  CRD$STOP, CRD$GB_CRD_DEBUG
                .EXTRN  CRD$PRINT, CRD$COMMON_INITIALIZATIONS
                .EXTRN  MEN$CNTLC_MENU, MEN$FTMTBL_INIT
                .EXTRN  MEN$PREPTBL_INIT
                .EXTRN  MEN$CNTLCINIT_TBL_INIT
                .EXTRN  MEN$CNTLC_TBL_INIT
                .EXTRN  CRD$GB_CURRENT_STATE_TABLE
                .EXTRN  CRD$INIT_GENERAL_COM_TABLE
                .EXTRN  MEN$INIT_MM_STATE_TABLE
                .EXTRN  MEN$INIT_TQ_STATE_TABLE
                .EXTRN  MEN$INIT_HS_STATE_TABLE

                .PSECT  CODE,NOWRT,  SHR,  PIC,2

                  0000 00000     .ENTRY  MEN$MENU_FNCTST_INIT, Save nothing       ; 0192
                11 00000000G  EF  E9 00002     BLBC    CRD$GB_CRD_DEBUG, 1$        ; 0236
00000000'       EF  9F 00009     PUSHAB  P.AAD           ; 0237
                01  DD 0000F     PUSHL   #1             ;
                1A  DD 00011     PUSHL   #26            ;
00000000G       FF      03  FB 00013     CALLS   #3, @CRD$PRINT          ;
                ;
                ;  Initialize flags
                ;
00000000'       EF  94 0001A 1$:    CLRB    MEN$GB_FNCTST_INIT          ; 0244
                ;
                ; Initialize things common to both Auto Test and Menu Test
                ;
00000000G       EF  9F 00020     PUSHAB  MEN$CNTLC_MENU          ; 0252
00000000G       EF      01  FB 00026     CALLS   #1, CRD$COMMON_INITIALIZATIONS       ;
0000FE00        9F      02  88 0002D     BISB2   #2, @#^X0000FE00            ; 0254
                50 04    AC  D0 00034     MOVL    DSA_EXTRACT, R0            ; 0256
                02       50  D1 00038     CMPL    R0, #2             ; 0258
                09  12 0003B     BNEQ    2$             ;
0000FE02        9F      01  88 0003D     BISB2   #1, @#^X0000FE02          ;
                17  11 00044     BRB     4$             ;
                04       50  D1 00046 2$:    CMPL    R0, #4             ; 0259
                09  12 00049     BNEQ    3$             ;
0000FE02        9F      04  88 0004B     BISB2   #4, @#^X0000FE02          ;
                09  11 00052     BRB     4$             ;
                0D  DD 00054 3$:    PUSHL   #13               ; 0260
00000000G       EF      01  FB 00056     CALLS   #1, CRD$STOP            ;
                ;
                ; Initialize Menu tables.
                ; Since the  CRD MENU TEST  control code is a loadable image,
                ; link-time  resolved  address  pointers  must have an offset
                ; added to them.  This  is  because  the address pointers are
                ; resolved at link time with reference to a zero base address
                ; However,  at  run  time,  the  code  will  be  loaded at an
                ; address starting at CRD$AL_CRD_Dispatch_Vectors. Therefore,
                ; for  all  tables with text address pointers,  add an offset
                ; equal to the address of CRD$AL_CRD_Dispatch_Vectors.
00000000G       EF      00  FB 0005D 4$:    CALLS   #0, MEN$FTMTBL_INIT          ; 0280
```

```
                00000000G  EF      00  FB 00064    CALLS   #0, MEN$PREPTBL_INIT          ; 0286
                00000000G  EF      00  FB 0006B    CALLS   #0, MEN$CNTLCINIT_TBL_INIT       ; 0292
                00000000G  EF      00  FB 00072    CALLS   #0, MEN$CNTLC_TBL_INIT         ; 0298
                   ;
                   ; Initialize the General Commands Table
                   ; and the Menu State tables:
                   ;
                00000000G  EF  94 00079    CLRB    CRD$GB_CURRENT_STATE_TABLE         ; 0307
                00000000G  EF      00  FB 0007F    CALLS   #0, CRD$INIT_GENERAL_COM_TABLE      ; 0309
                00000000G  EF      00  FB 00086    CALLS   #0, MEN$INIT_MM_STATE_TABLE        ; 0310
                00000000G  EF      00  FB 0008D    CALLS   #0, MEN$INIT_TQ_STATE_TABLE        ; 0311
                00000000G  EF      00  FB 00094    CALLS   #0, MEN$INIT_HS_STATE_TABLE        ; 0312
                   ;
                   ; Initialization complete
                   ;
                0000FE00   9F      02  88 0009B    BISB2   #2, @#^X0000FE00           ; 0320
                    50 04   AC  D0 000A2    MOVL    DSA_EXTRACT, R0          ; 0321
                    02      50  D1 000A6    CMPL    R0, #2           ; 0323
                    09  12 000A9    BNEQ    5$              ;
                0000FE02   9F      01  88 000AB    BISB2   #1, @#^X0000FE02           ;
                    17  11 000B2    BRB     7$              ;
                    04      50  D1 000B4 5$:   CMPL    R0, #4           ; 0324
                    09  12 000B7    BNEQ    6$              ;
                0000FE02   9F      04  88 000B9    BISB2   #4, @#^X0000FE02           ;
                    09  11 000C0    BRB     7$              ;
                    0D  DD 000C2 6$:   PUSHL   #13             ; 0325
                00000000G  EF      01  FB 000C4    CALLS   #1, CRD$STOP             ;
                    11 00000000G  EF  E9 000CB 7$:   BLBC    CRD$GB_CRD_DEBUG, 8$          ; 0329
                00000000'  EF  9F 000D2    PUSHAB  P.AAE           ; 0330
                    01  DD 000D8    PUSHL   #1              ;
                    1A  DD 000DA    PUSHL   #26             ;
                00000000G  FF      03  FB 000DC    CALLS   #3, @CRD$PRINT           ;
                    50      01  D0 000E3 8$:   MOVL    #1, R0           ; 0332
                 04 000E6    RET                 ; 0333

; Routine Size:  231 bytes,    Routine Base:  CODE + 0000


;   0334  1 %SBTTL 'MEN$Build_DDB - Build DDB Database Routine'
;   0335  1
;   0336  1 GLOBAL ROUTINE MEN$Build_DDB =
;   0337  1
;   0338  1 !++
;   0339  1 ! FUNCTIONAL DESCRIPTION:
;   0340  1 !
;   0341  1 !    Creates DDB database.  One DDB is built for each VDS P-Table.
;   0342  1 !
;   0343  1 ! INPUT PARAMETERS
;   0344  1 !
;   0345  1 ! OUTPUT PARAMETERS
;   0346  1 !
;   0347  1 ! IMPLICIT INPUTS
;   0348  1 !
;   0349  1 !    VDS P-Tables exist pointed to by address pointer located in CRD$GA_PTable
;   0350  1 !
```

```
;    0351  1 ! EXPLICIT OUTPUTS
;    0352  1 !
;    0353  1 !    Circular doubly-linked queue list of DDB's.
;    0354  1 !
;    0355  1 ! SIDE EFFECTS
;    0356  1 !
;    0357  1 !    On error this routine aborts CRD MENU by calling CRD$Stop.
;    0358  1 !
;    0359  1 ! COMPLETION CODES
;    0360  1 !
;    0361  1 !--
;    0362  2 BEGIN
;    0363  2    LOCAL
;    0364  2  DDB_Ptr   : REF Device_Data_Block_Structure,
;    0365  2  TstQ_Size,
;    0366  2  DDB_Size;
;    0367  2
;    0368  2    REGISTER
;    0369  2  DDB_TstQ_Tbl_Entries,
;    0370  2  DDB_Count,
;    0371  2  PTable_Table_Index,
;    0372  2  PTable;
;    0373  2
;    0374  2    BIND
;    0375  2  Numb_PTable_Entries = (.CRD$GA_PTable),
;    0376  2  PTable_Table     = (.CRD$GA_PTable+4) : VECTOR [,LONG];
;    0377  2
;    0378  2    IF (.CRD$GB_CRD_Debug) THEN
;    0379  2  $CRD_Print ($ASCIC ('MEN$Build_DDB: Starting to build the DDBs!/'));
;    0380  2
;    0381  2    !+
;    0382  2    ! Initialize the DDB Queue
;    0383  2    !-
;    0384  2
;    0385  2    MEN$GA_DDB_Head [Queue$L_Flink] = MEN$GA_DDB_Head;
;    0386  2    MEN$GA_DDB_Head [Queue$L_Blink] = MEN$GA_DDB_Head;
;    0387  2
;    0388  2    !+
;    0389  2    ! Get a valid VDS P-Table entry and build a DDB.
;    0390  2    ! Start at the last P-Table entry and work backwards
;    0391  2    !-
;    0392  2
;    0393  2    DDB_Count = 0;
;    0394  2    PTable_Table_Index = .Numb_PTable_Entries ;
;    0395  2
;    0396  2    WHILE ((PTable_Table_Index = .PTable_Table_Index-1) GEQ 0) DO
;    0397  3  BEGIN
;    0398  3  PTable = .PTable_Table [.PTable_Table_Index];
;    0399  3
;    0400  3  IF .PTable NEQ 0
;    0401  3  THEN
;    0402  4      BEGIN
;    0403  5      IF (NOT MEN$Allocate_Memory (%REF(CRD$K_DDB_Last_Entry*4), DDB_Ptr, DDB_Size))
;    0404  4      THEN
;    0405  4   Internal_Error_Exit (CRD$K_Allocation_Error, (CRD$K_DDB_Last_Entry * 4), .DDB_Size, .DDB_Ptr);
```

```
;    0406  4
;    0407  4       INSQUE (.DDB_Ptr, .MEN$GA_DDB_Head [Queue$L_Blink]);
;    0408  4       DDB_Ptr [CRD$K_DDB_PTable_Entry] = .PTable;
;    0409  4       DDB_Ptr [CRD$K_DDB_Memory_Size] = .DDB_Size;
;    0410  4       DDB_Count = .DDB_Count + 1;
;    0411  3       END;
;    0412  2   END;
;    0413  2
;    0414  2       !+
;    0415  2       ! Allocate enough memory for DDB Test Queue pointer table
;    0416  2       !-
;    0417  2
;    0418  2       DDB_TstQ_Tbl_Entries = .DDB_Count +1;
;    0419  2       ! Create one more entry for a null DDB pointer
;    0420  2
;    0421  3       IF (NOT MEN$ALLOCATE_Memory (%REF (.DDB_TstQ_Tbl_Entries * 4),
;    0422  3        MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr],
;    0423  3        TstQ_Size))
;    0424  2       THEN
; P  0425  2   Internal_Error_Exit (CRD$K_Allocation_Error, (.DDB_TstQ_Tbl_Entries * 4),
;    0426  2    .TstQ_Size, .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr]);
;    0427  2
;    0428  2       MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb_Entries] = (.TstQ_Size/4);
;    0429  2       ! number of longword entries
;    0430  2
;    0431  2       MEN$GA_Test_Queue [MEN$V_TQ_Table_Size] = .TstQ_Size;
;    0432  2       ! Size in bytes of allocated memory for table
;    0433  2
;    0434  2       IF (.CRD$GB_CRD_Debug) THEN
;    0435  2     $CRD_Print ($ASCIC ('MEN$Build_DDB: Done building the DDBs!/'));
;    0436  2
;    0437  2       RETURN (CRD$K_Success);
;    0438  1   END;



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

3A  42  44  44  5F  64  6C  69  75  42  24  4E  45  4D  2B  00087 P.AAF:  .ASCII  \+MEN$Build_DDB: Starting to build the DD\  ;
75  62  20  6F  74  20  67  6E  69  74  72  61  74  53  20  00096             ;
      44  44  20  65  68  74  20  64  6C  69  000A5             ;
          2F  21  73  42  000AF     .ASCII  \Bs!/\          ;
3A  42  44  44  5F  64  6C  69  75  42  24  4E  45  4D  27  000B3 P.AAG:  .ASCII  \'MEN$Build_DDB: Done building the DDBs!/\  ;
20  67  6E  69  64  6C  69  75  62  20  65  6E  6F  44  20  000C2             ;
      2F  21  73  42  44  44  20  65  68  74  000D1             ;

            .EXTRN  CRD$GA_PTABLE

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              003C 00000     .ENTRY  MEN$BUILD_DDB, Save R2,R3,R4,R5       ; 0336
            5E       10  C2 00002     SUBL2   #16, SP              ;
            53 00000000G   EF  D0 00005     MOVL    CRD$GA_PTABLE, R3          ; 0375
            55 00000000G   EF       04  C1 0000C     ADDL3   #4, CRD$GA_PTABLE, R5          ; 0376
            11 00000000G   EF  E9 00014     BLBC    CRD$GB_CRD_DEBUG, 1$          ; 0378
```

```
                00000000'  EF  9F 0001B    PUSHAB  P.AAF           ; 0379
                    01  DD 00021    PUSHL   #1              ;
                    1A  DD 00023    PUSHL   #26             ;
                00000000G  FF       03  FB 00025    CALLS   #3, @CRD$PRINT          ;
                00000000'  EF 00000000'  EF  9E 0002C 1$:   MOVAB   MEN$GA_DDB_HEAD, MEN$GA_DDB_HEAD       ; 0385
                00000000'  EF 00000000'  EF  9E 00037    MOVAB   MEN$GA_DDB_HEAD, MEN$GA_DDB_HEAD+4     ; 0386
                    52  D4 00042    CLRL    DDB_COUNT           ; 0393
                    53     63  D0 00044    MOVL    (R3), PTABLE_TABLE_INDEX        ; 0394
                    4F  11 00047    BRB     4$              ; 0396
                    54    6543  D0 00049 2$:   MOVL    (R5)[PTABLE_TABLE_INDEX], PTABLE        ; 0398
                    49  13 0004D    BEQL    4$              ; 0400
                04  AE  9F 0004F    PUSHAB  DDB_SIZE            ; 0403
                0C  AE  9F 00052    PUSHAB  DDB_PTR             ;
                08  AE     2C  D0 00055    MOVL    #44, 8(SP)          ;
                08  AE  9F 00059    PUSHAB  8(SP)               ;
                 00000000V  EF       03  FB 0005C    CALLS   #3, MEN$ALLOCATE_MEMORY         ;
                    1B     50  E8 00063    BLBS    R0, 3$              ;
                08  AE  DD 00066    PUSHL   DDB_PTR             ; 0405
                08  AE  DD 00069    PUSHL   DDB_SIZE            ;
                    2C  DD 0006C    PUSHL   #44             ;
                    7E     01  CE 0006E    MNEGL   #1, -(SP)           ;
                 00000000G  EF       04  FB 00071    CALLS   #4, MEN$MENU_TEST_INTERNAL_ERROR        ;
                    0D  DD 00078    PUSHL   #13             ;
                 00000000G  EF       01  FB 0007A    CALLS   #1, CRD$STOP            ;
                 00000000'  FF 08    BE  0E 00081 3$:   INSQUE  @DDB_PTR, @MEN$GA_DDB_HEAD+4         ; 0407
                    50 08    AE  D0 00089    MOVL    DDB_PTR, R0             ; 0408
                08  A0      54  D0 0008D    MOVL    PTABLE, 8(R0)           ;
                24  A0 04    AE  D0 00091    MOVL    DDB_SIZE, 36(R0)            ; 0409
                    52  D6 00096    INCL    DDB_COUNT           ; 0410
                    AE      53  F4 00098 4$:   SOBGEQ  PTABLE_TABLE_INDEX, 2$          ; 0396
                    50 01    A2  9E 0009B    MOVAB   1(R2), DDB_TSTQ_TBL_ENTRIES         ; 0418
                0C  AE  9F 0009F    PUSHAB  TSTQ_SIZE           ; 0421
                 00000000'  EF  9F 000A2    PUSHAB  MEN$GA_TEST_QUEUE+4         ; 0422
                    52        50       02  78 000A8    ASHL    #2, DDB_TSTQ_TBL_ENTRIES, R2        ; 0421
                08  AE     52  D0 000AC    MOVL    R2, 8(SP)           ;
                08  AE  9F 000B0    PUSHAB  8(SP)               ;
                 00000000V  EF       03  FB 000B3    CALLS   #3, MEN$ALLOCATE_MEMORY         ;
                    1E     50  E8 000BA    BLBS    R0, 5$              ;
                 00000000'  EF  DD 000BD    PUSHL   MEN$GA_TEST_QUEUE+4         ; 0426
                10  AE  DD 000C3    PUSHL   TSTQ_SIZE           ;
                    52  DD 000C6    PUSHL   R2              ;
                    7E     01  CE 000C8    MNEGL   #1, -(SP)           ;
                 00000000G  EF       04  FB 000CB    CALLS   #4, MEN$MENU_TEST_INTERNAL_ERROR        ;
                    0D  DD 000D2    PUSHL   #13             ;
                 00000000G  EF       01  FB 000D4    CALLS   #1, CRD$STOP            ;
                 00000000'  EF    0C    AE       04  C7 000DB 5$:   DIVL3   #4, TSTQ_SIZE, MEN$GA_TEST_QUEUE       ; 0428
                 00000000'  EF 0C    AE  D0 000E4    MOVL    TSTQ_SIZE, MEN$GA_TEST_QUEUE+8         ; 0431
                    11 00000000G  EF  E9 000EC    BLBC    CRD$GB_CRD_DEBUG, 6$            ; 0434
                 00000000'  EF  9F 000F3    PUSHAB  P.AAG           ; 0435
                    01  DD 000F9    PUSHL   #1              ;
                    1A  DD 000FB    PUSHL   #26             ;
                 00000000G  FF       03  FB 000FD    CALLS   #3, @CRD$PRINT          ;
                    50     01  D0 00104 6$:   MOVL    #1, R0              ; 0437
                     04 00107    RET                 ; 0438
```

```
; Routine Size:  264 bytes,    Routine Base:  CODE + 00E7


;    0439  1 %SBTTL 'MEN$Determine_Support Routine'
;    0440  1
;    0441  1 GLOBAL ROUTINE MEN$Determine_Support =
;    0442  1
;    0443  1 !++
;    0444  1 ! FUNCTIONAL DESCRIPTION:
;    0445  1 !
;    0446  1 !    Loads the support database file into memory and for each DDB
;    0447  1 !    searches the support database for a support block.  If a suppport block
;    0448  1 !    is found then hardware support tables are created.
;    0449  1 !
;    0450  1 ! INPUT PARAMETERS
;    0451  1 !
;    0452  1 ! OUTPUT PARAMETERS
;    0453  1 !
;    0454  1 ! IMPLICIT INPUTS
;    0455  1 !
;    0456  1 ! EXPLICIT OUTPUTS
;    0457  1 !
;    0458  1 ! SIDE EFFECTS
;    0459  1 !
;    0460  1 !    On error this routine aborts CRD MENU by calling CRD$Stop.
;    0461  1 !
;    0462  1 ! COMPLETION CODES
;    0463  1 !
;    0464  1 !    CRD$K_Success
;    0465  1 !--
;    0466  2 BEGIN
;    0467  2     LOCAL
;    0468  2  SupBlk_Ptr   : REF $MEN_SupBlk_ATTR;
;    0469  2
;    0470  2     REGISTER
;    0471  2  DatBase_ID : REF $MEN_DBaseID_ATTR,
;    0472  2  Device_Name_Ptr,
;    0473  2  DDB_Ptr   : REF Device_Data_Block_Structure,
;    0474  2  PTable_Ptr   : REF $DS_HPO_DECL ();
;    0475  2
;    0476  2     IF (.CRD$GB_CRD_Debug) THEN
;    0477  2  $CRD_Print ($ASCIC ('MEN$Determine_Support: Starting to determine the Hardware Support!/'));
;    0478  2
;    0479  2     !+
;    0480  2     ! Build filename ASCID's
;    0481  2     !-
;    0482  2
;    0483  2     A_MenFuncSup_File [DSC$W_Length]  = .MEN$GT_MenFuncSup_File [0];
;    0484  2     A_MenFuncSup_File [DSC$A_Pointer] = MEN$GT_MenFuncSup_File [1];
;    0485  2
;    0486  2     A_File_Default [DSC$W_Length]     = .MEN$GT_File_Default [0];
;    0487  2     A_File_Default [DSC$A_Pointer]    = MEN$GT_File_Default [1];
;    0488  2
;    0489  2     !+
;    0490  2     ! Load Support File database into memory and initialize support
```

```
;    0491  2      ! database descriptor
;    0492  2      !-
;    0493  2
;    0494  2      MEN$Load_File  (A_MenFuncSup_File,
;    0495  2        A_File_Default,
;    0496  2        MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Size],
;    0497  2        MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Start],
;    0498  2        MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Total_Siz]);
;    0499  2
;    0500  2      DatBase_ID = .MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Start];
;    0501  2      ! Point to the loaded database identification
;    0502  2      ! block
;    0503  2
;    0504  2      MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Ptr] =
;    0505  2   ((.DatBase_ID + 4) + (.DatBase_ID [DBaseID$L_Size] * 4));
;    0506  2      ! Point to the first device support block
;    0507  2
;    0508  2      !+
;    0509  2      ! For each DDB search for a device support block in the
;    0510  2      ! support database
;    0511  2      !-
;    0512  2
;    0513  2      DDB_Ptr = MEN$GA_DDB_Head;
;    0514  2
;    0515  2      WHILE ((DDB_Ptr = .DDB_Ptr [CRD$K_DDB_Flink]) NEQ MEN$GA_DDB_Head) DO
;    0516  3    BEGIN
;    0517  3    PTable_Ptr = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    0518  3    Device_Name_Ptr = PTable_Ptr [HP$T_Type];
;    0519  3
;    0520  3    !+
;    0521  3    ! If a support block exists then load DDB info.
;    0522  3    !-
;    0523  3
;    0524  4    IF (MEN$Find_SupBlk (MEN$GA_FuncSup_DatBase,
;    0525  4           .Device_Name_Ptr,
;    0526  4           SupBlk_Ptr))
;    0527  3    THEN
;    0528  4        BEGIN
;    0529  4        DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr] = .SupBlk_Ptr;
;    0530  4
;    0531  4        !+
;    0532  4        ! Create Hardware Support Table for the device
;    0533  4        !-
;    0534  4
;    0535  5        IF (NOT Create_HST (.DDB_Ptr))
;    0536  4        THEN
;    0537  4     Internal_Error_Exit (CRD$K_Create_HST, 0, 0, .DDB_Ptr);
;    0538  3        END;
;    0539  2    END;
;    0540  2
;    0541  2      IF (.CRD$GB_CRD_Debug) THEN
;    0542  2    $CRD_Print ($ASCIC ('MEN$Determine_Support: Done determining the Hardware Support!/'));
;    0543  2
;    0544  2      RETURN (CRD$K_Success);
;    0545  1  END;
```

```
                .PSECT  DATA,NOWRT,NOEXE,  SHR,2

5F  65  6E  69  6D  72  65  74  65  44  24  4E  45  4D  43  000DB P.AAH:  .ASCII  \CMEN$Determine_Support: Starting to dete\  ;
69  74  72  61  74  53  20  3A  74  72  6F  70  70  75  53  000EA             ;
      65  74  65  64  20  6F  74  20  67  6E  000F9                          ;
77  64  72  61  48  20  65  68  74  20  65  6E  69  6D  72  00103    .ASCII  \rmine the Hardware Support!/\       ;
 2F  21  74  72  6F  70  70  75  53  20  65  72  61  00112                   ;
5F  65  6E  69  6D  72  65  74  65  44  24  4E  45  4D  3E  0011F P.AAI:  .ASCII  \>MEN$Determine_Support: Done determining\  ;
64  20  65  6E  6F  44  20  3A  74  72  6F  70  70  75  53  0012E             ;
      67  6E  69  6E  69  6D  72  65  74  65  0013D                          ;
53  20  65  72  61  77  64  72  61  48  20  65  68  74  20  00147    .ASCII  \ the Hardware Support!/\       ;
       2F  21  74  72  6F  70  70  75  00156                 ;


                .PSECT  CODE,NOWRT,  SHR,  PIC,2

                  001C 00000     .ENTRY  MEN$DETERMINE_SUPPORT, Save R2,R3,R4       ; 0441
                5E        04  C2 00002     SUBL2   #4, SP                ;
                11 00000000G  EF  E9 00005     BLBC    CRD$GB_CRD_DEBUG, 1$         ; 0476
            00000000'  EF  9F 0000C     PUSHAB  P.AAH                 ; 0477
                01  DD 00012     PUSHL   #1                  ;
                1A  DD 00014     PUSHL   #26                  ;
            00000000G  FF       03  FB 00016     CALLS   #3, @CRD$PRINT              ;
            00000000'  EF 00000000'  EF  9B 0001D 1$:   MOVZBW  MEN$GT_MENFUNCSUP_FILE, A_MENFUNCSUP_FILE    ; 0483
            00000000'  EF 00000000'  EF  9E 00028     MOVAB   MEN$GT_MENFUNCSUP_FILE+1, -          ; 0484
                A_MENFUNCSUP_FILE+4          ;
            00000000'  EF 00000000'  EF  9B 00033     MOVZBW  MEN$GT_FILE_DEFAULT, A_FILE_DEFAULT        ; 0486
            00000000'  EF 00000000'  EF  9E 0003E     MOVAB   MEN$GT_FILE_DEFAULT+1, A_FILE_DEFAULT+4     ; 0487
            00000000'  EF  9F 00049     PUSHAB  MEN$GA_FUNCSUP_DATBASE+8          ; 0498
            00000000'  EF  9F 0004F     PUSHAB  MEN$GA_FUNCSUP_DATBASE+12          ; 0497
            00000000'  EF  9F 00055     PUSHAB  MEN$GA_FUNCSUP_DATBASE           ; 0496
            00000000'  EF  9F 0005B     PUSHAB  A_FILE_DEFAULT           ; 0494
            00000000'  EF  9F 00061     PUSHAB  A_MENFUNCSUP_FILE           ;
            00000000V  EF       05  FB 00067     CALLS   #5, MEN$LOAD_FILE           ;
                50 00000000'  EF  D0 0006E     MOVL    MEN$GA_FUNCSUP_DATBASE+12, DATBASE_ID        ; 0500
                51        60  D0 00075     MOVL    (DATBASE_ID), R1           ; 0505
            00000000'  EF 04 A041  DE 00078     MOVAL   4(DATBASE_ID)[R1], MEN$GA_FUNCSUP_DATBASE+4 ;
                53 00000000'  EF  9E 00081     MOVAB   MEN$GA_DDB_HEAD, DDB_PTR          ; 0513
                43  11 00088     BRB     3$                  ; 0515
                54 08    A3  D0 0008A 2$:   MOVL    8(DDB_PTR), PTABLE_PTR          ; 0517
                52 26    A4  9E 0008E     MOVAB   38(R4), DEVICE_NAME_PTR          ; 0518
                4004    8F  BB 00092     PUSHR   #^M<R2,SP>              ; 0525
            00000000'  EF  9F 00096     PUSHAB  MEN$GA_FUNCSUP_DATBASE           ; 0524
            00000000V  EF       03  FB 0009C     CALLS   #3, MEN$FIND_SUPBLK           ;
                27        50  E9 000A3     BLBC    R0, 3$                ;
            20  A3        6E  D0 000A6     MOVL    SUPBLK_PTR, 32(DDB_PTR)          ; 0529
                53  DD 000AA     PUSHL   DDB_PTR                ; 0535
            00000000V  EF       01  FB 000AC     CALLS   #1, CREATE_HST           ;
                17        50  E8 000B3     BLBS    R0, 3$                ;
                53  DD 000B6     PUSHL   DDB_PTR                ; 0537
                7E  7C 000B8     CLRQ    -(SP)                ;
```

```
                7E      0B  CE 000BA    MNEGL   #11, -(SP)             ;
            00000000G   EF      04  FB 000BD    CALLS   #4, MEN$MENU_TEST_INTERNAL_ERROR       ;
                0D  DD 000C4    PUSHL   #13             ;
            00000000G   EF      01  FB 000C6    CALLS   #1, CRD$STOP            ;
                53      63  D0 000CD 3$:   MOVL    (DDB_PTR), DDB_PTR          ; 0515
                50 00000000'    EF  9E 000D0    MOVAB   MEN$GA_DDB_HEAD, R0         ;
                50      53  D1 000D7    CMPL    DDB_PTR, R0             ;
                AE  12 000DA    BNEQ    2$              ;
                11 00000000G    EF  E9 000DC    BLBC    CRD$GB_CRD_DEBUG, 4$            ; 0541
            00000000'   EF  9F 000E3    PUSHAB  P.AAI               ; 0542
                01  DD 000E9    PUSHL   #1              ;
                1A  DD 000EB    PUSHL   #26             ;
            00000000G   FF      03  FB 000ED    CALLS   #3, @CRD$PRINT          ;
                50      01  D0 000F4 4$:   MOVL    #1, R0              ; 0544
             04 000F7    RET                 ; 0545

; Routine Size:  248 bytes,    Routine Base:  CODE + 01EF


;    0546  1 %SBTTL 'Create_HST Routine'
;    0547  1
;    0548  1 ROUTINE Create_HST ( I_DDB)  =
;    0549  1
;    0550  1 !++
;    0551  1 ! FUNCTIONAL DESCRIPTION:
;    0552  1 !
;    0553  1 !    For the DDB specified create Hardware Support Table(s).  One Hardware
;    0554  1 !    Support Table is built for each Diagnostic Block found in the DDB's Support
;    0555  1 !    Block.
;    0556  1 !
;    0557  1 ! INPUT PARAMETERS
;    0558  1 !
;    0559  1 !    I_DDB  Address of DDB
;    0560  1 !
;    0561  1 ! OUTPUT PARAMETERS
;    0562  1 !
;    0563  1 ! IMPLICIT INPUTS
;    0564  1 !
;    0565  1 !    The DDB points to a support block
;    0566  1 !
;    0567  1 ! EXPLICIT OUTPUTS
;    0568  1 !
;    0569  1 !    The DDB shall point to one or more concatenated Hardware Support Tables.
;    0570  1 ! SIDE EFFECTS
;    0571  1 !
;    0572  1 !
;    0573  1 ! COMPLETION CODES
;    0574  1 !
;    0575  1 ! CRD$K_Success
;    0576  1 ! CRD$K_Failure
;    0577  1 !--
;    0578  2 BEGIN
;    0579  2     LOCAL
;    0580  2  HST_Size,
;    0581  2  HST_Ptr  : REF VECTOR [, LONG];
```

```
;    0582  2
;    0583  2     REGISTER
;    0584  2  Record_No,
;    0585  2  Record_Cnt,
;    0586  2  Record_Ptr : REF VECTOR [, BYTE],
;    0587  2
;    0588  2  DiagBlk_No,
;    0589  2  DiagBlk_Index,
;    0590  2  DiagBlk_Ptr  : REF $MEN_DiagBlk_ATTR,
;    0591  2
;    0592  2  TstCnfg_Ptr  : REF $MEN_TstCnfg_ATTR,
;    0593  2  SupBlk_Ptr  : REF $MEN_SupBlk_ATTR,
;    0594  2  DDB_Ptr  : REF Device_Data_Block_Structure;
;    0595  2
;    0596  2     BIND
;    0597  2  DDB = .I_DDB;
;    0598  2
;    0599  2     DDB_Ptr = DDB;  ! Point to DDB
;    0600  2     SupBlk_Ptr = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr];
;    0601  2     ! Point to the support block
;    0602  2
;    0603  2     TstCnfg_Ptr = SupBlk_Ptr [SupBlk$B_StrtFrst_CnfgBlk];
;    0604  2     ! Point to the first configuration block
;    0605  2
;    0606  2     DiagBlk_No = (.TstCnfg_Ptr [TstCnfg$B_DiagBlk_No]) ;
;    0607  2     ! Specify number of diagnostic blocks
;    0608  2
;    0609  2     !+
;    0610  2     ! Allocate memory space for Hardware Support Table.  The memory space
;    0611  2     ! should accomodate 1 or more concatenated HST's
;    0612  2     !-
;    0613  2
;    0614  3     IF (NOT MEN$Allocate_Memory (%REF (CRD$K_Last_Entry * 4 * .DiagBlk_No), HST_Ptr, HST_Size))
;    0615  2     THEN
;    0616  2  RETURN (CRD$K_Failure);
;    0617  2
;    0618  2     DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] = .HST_Ptr;
;    0619  2     ! Point to Hardware Support Table
;    0620  2
;    0621  2     DDB_Ptr [CRD$K_DDB_Menu_Support_Size] = .HST_Size;
;    0622  2     ! Size in bytes of allocated memory
;    0623  2
;    0624  2     DDB_Ptr [CRD$K_DDB_Numb_Support_Tables] = .TstCnfg_Ptr [TstCnfg$B_DiagBlk_No];
;    0625  2     ! Specify the number of concatenated
;    0626  2     ! Hardware Support Tables to be built
;    0627  2
;    0628  2     !+
;    0629  2     ! Setup for creating Hardware Support Table from Diagnostic Block
;    0630  2     ! information
;    0631  2     !-
;    0632  2
;    0633  2     DiagBlk_No = (.TstCnfg_Ptr [TstCnfg$B_DiagBlk_No]) ;
;    0634  2     ! Specify number of diagnostic blocks
;    0635  2
;    0636  2     DiagBlk_Index = - 1; ! Set up the Diagnostic Block index
```

```
;    0637  2
;    0638  2     DiagBlk_Ptr = TstCnfg_Ptr [TstCnfg$B_StrtFrst_DiagBlk];
;    0639  2     ! Point to the first diagnostic block
;    0640  2
;    0641  2     !+
;    0642  2     ! Start loading Hardware Support Blocks
;    0643  2     ! *** NOTE ***
;    0644  2     !    The sequence of loading ASCIC record pointer  information into
;    0645  2     !    each Hardware Support Table
;    0646  2     !    must duplicate the sequence in which the information
;    0647  2     !    is found in the Diagnostic Block.  Also the number of times the
;    0648  2     !    record pointer is updated must equal the number of records in the Diagnostic
;    0649  2     !    Block in order to correctly sequence through the Diagnostic Blocks.
;    0650  2     !-
;    0651  2
;    0652  2     WHILE ( (DiagBlk_Index = .DiagBlk_Index + 1) LSS .DiagBlk_No ) DO
;    0653  3  BEGIN
;    0654  3  Record_Cnt = 0;
;    0655  3  Record_No = .DiagBlk_Ptr [DiagBlk$W_Record_No];
;    0656  3     ! Number of records in Diagnostic Block
;    0657  3  Record_Ptr = DiagBlk_Ptr [DiagBlk$B_Frst_Record];
;    0658  3
;    0659  3  !+
;    0660  3  ! load Diagnostic Image Name
;    0661  3  !-
;    0662  3
;    0663  3  HST_Ptr [CRD$K_L agnostic_Name] = .Record_Ptr;
;    0664  3  Record_Ptr = .Record_Ptr [0] + Record_Ptr [1];
;    0665  3  Record_Cnt = .Record_Cnt + 1;
;    0666  3
;    0667  3  !+
;    0668  3  ! load VDS SET FLAGS command arguements ASCIC pointer
;    0669  3  !-
;    0670  3
;    0671  3  HST_Ptr [CRD$K_Set_Flags_Args] = .Record_Ptr;
;    0672  3  Record_Ptr = .Record_Ptr [0] + Record_Ptr [1];
;    0673  3  Record_Cnt = .Record_Cnt + 1;
;    0674  3
;    0675  3  !+
;    0676  3  ! load VDS SET EVENT command arguements ASCIC pointer
;    0677  3  !-
;    0678  3
;    0679  3  HST_Ptr [CRD$K_Set_Event_Args] = .Record_Ptr;
;    0680  3  Record_Ptr = .Record_Ptr [0] + Record_Ptr [1];
;    0681  3  Record_Cnt = .Record_Cnt + 1;
;    0682  3
;    0683  3  !+
;    0684  3  ! load VDS START command qualifiers ASCIC pointer
;    0685  3  !-
;    0686  3
;    0687  3  HST_Ptr [CRD$K_Start_Qualifiers] = .Record_Ptr;
;    0688  3  Record_Ptr = .Record_Ptr [0] + Record_Ptr [1];
;    0689  3  Record_Cnt = .Record_Cnt + 1;
;    0690  3
;    0691  3  !+
```

```
;    0692  3  ! load Test Start text test class field ASCIC pointer
;    0693  3  !-
;    0694  3
;    0695  3  HST_Ptr [CRD$K_Test_Start_Text] = .Record_Ptr;
;    0696  3  Record_Ptr = .Record_Ptr [0] + Record_Ptr [1];
;    0697  3  Record_Cnt = .Record_Cnt + 1;
;    0698  3
;    0699  3  !+
;    0700  3  ! load run time ASCIC pointer
;    0701  3  !-
;    0702  3
;    0703  3  HST_Ptr [CRD$K_RunTime] = .Record_Ptr;
;    0704  3  Record_Ptr = .Record_Ptr [0] + Record_Ptr [1];
;    0705  3  Record_Cnt = .Record_Cnt + 1;
;    0706  3
;    0707  3  DiagBlk_Ptr = .Record_Ptr;
;    0708  3     ! Point to next Diagnostic Block location
;    0709  3  HST_Ptr = HST_Ptr [CRD$K_Last_Entry];
;    0710  3     ! Point ot the next Hardware Support Table
;    0711  3      ! location
;    0712  2  END;
;    0713  2
;    0714  2     RETURN (CRD$K_Success);
;    0715  1 END;



              00FC 00000 CREATE_HST:
            .WORD   Save R2,R3,R4,R5,R6,R7          ; 0548
            5E      0C  C2 00002    SUBL2   #12, SP          ;
            54 04   AC  D0 00005    MOVL    I_DDB, DDB_PTR          ; 0599
            50 20   A4  D0 00009    MOVL    32(DDB_PTR), SUPBLK_PTR         ; 0600
            52 17   A0  9E 0000D    MOVAB   23(R0), TSTCNFG_PTR         ; 0603
            53 01   A2  9A 00011    MOVZBL  1(TSTCNFG_PTR), DIAGBLK_NO          ; 0606
04          AE  9F 00015    PUSHAB  HST_SIZE            ; 0614
0C          AE  9F 00018    PUSHAB  HST_PTR          ;
08          AE      53      05  78 0001B    ASHL    #5, DIAGBLK_NO, 8(SP)          ;
08          AE  9F 00020    PUSHAB  8(SP)           ;
 00000000V  EF      03  FB 00023    CALLS   #3, MEN$ALLOCATE_MEMORY         ;
            03      50  E8 0002A    BLBS    R0, 1$          ;
          0091  31 0002D    BRW     4$              ;
10          A4 08   AE  D0 00030 1$:   MOVL    HST_PTR, 16(DDB_PTR)          ; 0618
28          A4 04   AE  D0 00035    MOVL    HST_SIZE, 40(DDB_PTR)           ; 0621
18          A4 01   A2  9A 0003A    MOVZBL  1(TSTCNFG_PTR), 24(DDB_PTR)          ; 0624
            53 01   A2  9A 0003F    MOVZBL  1(TSTCNFG_PTR), DIAGBLK_NO          ; 0633
            54      01  CE 00043    MNEGL   #1, DIAGBLK_INDEX           ; 0636
            55 04   A2  9E 00046    MOVAB   4(R2), DIAGBLK_PTR          ; 0638
            6A  11 0004A    BRB     3$              ; 0652
            51  D4 0004C 2$:   CLRL    RECORD_CNT          ; 0654
            50      65  3C 0004E    MOVZWL  (DIAGBLK_PTR), RECORD_NO          ; 0655
            52 02   A5  9E 00051    MOVAB   2(R5), RECORD_PTR           ; 0657
            56 08   AE  D0 00055    MOVL    HST_PTR, R6         ; 0663
            66      52  D0 00059    MOVL    RECORD_PTR, (R6)          ;
```

```
                57      62  9A 0005C    MOVZBL  (RECORD_PTR), R7          ; 0664
                57      52  C0 0005F    ADDL2   RECORD_PTR, R7          ;
                52 01   A7  9E 00062    MOVAB   1(R7), RECORD_PTR         ;
            04  A6      52  D0 00066    MOVL    RECORD_PTR, 4(R6)         ; 0671
                57      62  9A 0006A    MOVZBL  (RECORD_PTR), R7          ; 0672
                57      52  C0 0006D    ADDL2   RECORD_PTR, R7          ;
                52 01   A7  9E 00070    MOVAB   1(R7), RECORD_PTR         ;
            08  A6      52  D0 00074    MOVL    RECORD_PTR, 8(R6)         ; 0679
                57      62  9A 00078    MOVZBL  (RECORD_PTR), R7          ; 0680
                57      52  C0 0007B    ADDL2   RECORD_PTR, R7          ;
                52 01   A7  9E 0007E    MOVAB   1(R7), RECORD_PTR         ;
            10  A6      52  D0 00082    MOVL    RECORD_PTR, 16(R6)        ; 0687
                57      62  9A 00086    MOVZBL  (RECORD_PTR), R7          ; 0688
                57      52  C0 00089    ADDL2   RECORD_PTR, R7          ;
                52 01   A7  9E 0008C    MOVAB   1(R7), RECORD_PTR         ;
            14  A6      52  D0 00090    MOVL    RECORD_PTR, 20(R6)        ; 0695
                57      62  9A 00094    MOVZBL  (RECORD_PTR), R7          ; 0696
                57      52  C0 00097    ADDL2   RECORD_PTR, R7          ;
                52 01   A7  9E 0009A    MOVAB   1(R7), RECORD_PTR         ;
            18  A6      52  D0 0009E    MOVL    RECORD_PTR, 24(R6)        ; 0703
                56      62  9A 000A2    MOVZBL  (RECORD_PTR), R6          ; 0704
                56      52  C0 000A5    ADDL2   RECORD_PTR, R6          ;
                52 01   A6  9E 000A8    MOVAB   1(R6), RECORD_PTR         ;
                51      06  C0 000AC    ADDL2   #6, RECORD_CNT            ; 0705
                55      52  D0 000AF    MOVL    RECORD_PTR, DIAGBLK_PTR           ; 0707
            08  AE      20  C0 000B2    ADDL2   #32, HST_PTR              ; 0709
                54  D6 000B6 3$:   INCL    DIAGBLK_INDEX             ; 0652
                53      54  D1 000B8    CMPL    DIAGBLK_INDEX, DIAGBLK_NO         ;
                8F  19 000BB    BLSS    2$                ;
                50      01  D0 000BD    MOVL    #1, R0              ; 0714
            04 000C0    RET                     ;
                50  D4 000C1 4$:   CLRL    R0                ; 0715
            04 000C3    RET                     ;

; Routine Size:  196 bytes,    Routine Base:  CODE + 02E7


;   0716  1 %SBTTL 'MEN$Specify_SelectNo Routine'
;   0717  1
;   0718  1 GLOBAL ROUTINE MEN$Specify_SelectNo =
;   0719  1
;   0720  1 !++
;   0721  1 ! FUNCTIONAL DESCRIPTION:
;   0722  1 !
;   0723  1 ! INPUT PARAMETERS
;   0724  1 !
;   0725  1 ! OUTPUT PARAMETERS
;   0726  1 !
;   0727  1 ! IMPLICIT INPUTS
;   0728  1 !
;   0729  1 ! EXPLICIT OUTPUTS
;   0730  1 !
;   0731  1 ! SIDE EFFECTS
;   0732  1 !
;   0733  1 ! COMPLETION CODES
```

```
;    0734  1 !
;    0735  1 !--
```

```
;    0736  1
;    0737  2 BEGIN
;    0738  2  ROUTINE SelectNo_Class ( R_TEST_CLASS) : NOVALUE =
;    0739  3  BEGIN
;    0740  3  LOCAL
;    0741  3   SupBlk_Ptr : REF $MEN_Supblk_ATTR;
;    0742  3
;    0743  3  LOCAL
;    0744  3   Select_No_Index,
;    0745  3   DDB_Ptr  : REF Device_Data_Block_Structure,
;    0746  3   PTable_Ptr  : REF $DS_HPO_DECL ();
;    0747  3
;    0748  3  BIND
;    0749  3   Test_Class   = .R_TEST_CLASS,
;    0750  3   Support_Database = MEN$GA_FuncSup_DatBase : $MEN_FuncSup_DatBase_ATTR;
;    0751  3
;    0752  3  !+
;    0753  3  ! For each DDB of the specified Device Test Class, specify a
;    0754  3  ! unique Device Select Number
;    0755  3  !-
;    0756  3
;    0757  3  Select_No_Index = MEN$K_SelDev_Numb_Start;
;    0758  3  DDB_Ptr = MEN$GA_DDB_Head;
;    0759  3  WHILE ((DDB_Ptr = .DDB_Ptr [CRD$K_DDB_Flink]) NEQ MEN$GA_DDB_Head) DO
;    0760  4   BEGIN
;    0761  4   PTable_Ptr = .DDB_Ptr [CRD$K_DDB_PTable_Entry];
;    0762  4   SupBlk_Ptr = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr];
;    0763  5   IF (.SupBlk_Ptr NEQ 0)
;    0764  4   THEN
;    0765  5    BEGIN
;    0766  6    IF (.SupBlk_Ptr [SupBlk$B_Test_Class] EQL .Test_Class)
;    0767  5    THEN
;    0768  6     BEGIN
;    0769  6     DDB_Ptr [CRD$K_DDB_Menu_Device_Numb] = .Select_No_Index;
;    0770  6     Select_No_Index = .Select_No_Index + 1;
;    0771  5     END;
;    0772  4    END;
;    0773  3   END;
;    0774  2  END;    ! Routine SelectNo_Class


        SUPPORT_DATABASE=   MEN$GA_FUNCSUP_DATBASE


          001C 00000 SELECTNO_CLASS:
        .WORD   Save R2,R3,R4           ; 0738
        53      01  D0 00002    MOVL    #1, SELECT_NO_INDEX          ; 0757
        50 00000000'  EF  9E 00005    MOVAB   MEN$GA_DDB_HEAD, DDB_PTR         ; 0758
        17  11 0000C    BRB     2$              ; 0759
        54 08   A0  D0 0000E 1$:   MOVL    8(DDB_PTR), PTABLE_PTR         ; 0761
        52 20   A0  D0 00012    MOVL    32(DDB_PTR), SUPBLK_PTR         ; 0762
        0D  13 00016    BEQL    2$              ; 0763
    04  BC      11  A2      08      00  ED 00018    CMPZV   #0, #8, 17(SUPBLK_PTR), aR_TEST_CLASS       ; 0766
        04  12 0001F    BNEQ    2$              ;
```

```
      1C   A0       83  9E 00021    MOVAB   (SELECT_NO_INDEX)+, 28(DDB_PTR)       ; 0769
           50       60  D0 00025 2$:   MOVL    (DDB_PTR), DDB_PTR                 ; 0759
           51 00000000'  EF  9E 00028    MOVAB   MEN$GA_DDB_HEAD, R1                ;
           51       50  D1 0002F    CMPL    DDB_PTR, R1                ;
           DA  12 00032    BNEQ    1$                 ;
        04 00034    RET                 ; 0774

; Routine Size:  53 bytes,    Routine Base:  CODE + 03AB


;    0775  2
;    0776  2     SelectNo_Class ( %REF(MEN$K_TstCla_CPU));
;    0777  2     SelectNo_Class ( %REF(MEN$K_TstCla_Memory));
;    0778  2     SelectNo_Class ( %REF(MEN$K_TstCla_Disk));
;    0779  2     SelectNo_Class ( %REF(MEN$K_TstCla_Tape));
;    0780  2     SelectNo_Class ( %REF(MEN$K_TstCla_Comm));
;    0781  2     SelectNo_Class ( %REF(MEN$K_TstCla_Real));
;    0782  2     SelectNo_Class ( %REF(MEN$K_TstCla_Term));
;    0783  2     SelectNo_Class ( %REF(MEN$K_TstCla_Printer));
;    0784  2     SelectNo_Class ( %REF(MEN$K_TstCla_Card));
;    0785  2     SelectNo_Class ( %REF(MEN$K_TstCla_IO_Adaptor));
;    0786  2
;    0787  2     RETURN (CRD$K_Success);
;    0788  1 END;     ! Routine MEN$Specify_SelectNo



        0000 00000     .ENTRY  MEN$SPECIFY_SELECTNO, Save nothing       ; 0718
     01   DD 00002     PUSHL   #1              ; 0776
     5E   DD 00004     PUSHL   SP                 ;
C1   AF       01  FB 00006     CALLS   #1, SELECTNO_CLASS          ;
     6E       02  D0 0000A     MOVL    #2, (SP)           ; 0777
     5E   DD 0000D     PUSHL   SP                 ;
B8   AF       01  FB 0000F     CALLS   #1, SELECTNO_CLASS          ;
     6E       03  D0 00013     MOVL    #3, (SP)           ; 0778
     5E   DD 00016     PUSHL   SP                 ;
AF   AF       01  FB 00018     CALLS   #1, SELECTNO_CLASS          ;
     6E       04  D0 0001C     MOVL    #4, (SP)           ; 0779
     5E   DD 0001F     PUSHL   SP                 ;
A6   AF       01  FB 00021     CALLS   #1, SELECTNO_CLASS          ;
     6E       05  D0 00025     MOVL    #5, (SP)           ; 0780
     5E   DD 00028     PUSHL   SP                 ;
9D   AF       01  FB 0002A     CALLS   #1, SELECTNO_CLASS          ;
     6E       06  D0 0002E     MOVL    #6, (SP)           ; 0781
     5E   DD 00031     PUSHL   SP                 ;
94   AF       01  FB 00033     CALLS   #1, SELECTNO_CLASS          ;
     6E       07  D0 00037     MOVL    #7, (SP)           ; 0782
     5E   DD 0003A     PUSHL   SP                 ;
8B   AF       01  FB 0003C     CALLS   #1, SELECTNO_CLASS          ;
     6E       08  D0 00040     MOVL    #8, (SP)           ; 0783
     5E   DD 00043     PUSHL   SP                 ;
82   AF       01  FB 00045     CALLS   #1, SELECTNO_CLASS          ;
     6E       09  D0 00049     MOVL    #9, (SP)           ; 0784
```

```
            5E  DD 0004C     PUSHL   SP                      ;
    FF78    CF       01  FB 0004E    CALLS   #1, SELECTNO_CLASS          ;
            6E       0A  D0 00053    MOVL    #10, (SP)           ; 0785
            5E  DD 00056     PUSHL   SP                      ;
    FF6E    CF       01  FB 00058    CALLS   #1, SELECTNO_CLASS          ;
            50       01  D0 0005D    MOVL    #1, R0              ; 0787
         04 00060    RET                 ; 0788

; Routine Size:  97 bytes,    Routine Base:  CODE + 03E0


;    0789  1 %SBTTL 'MEN$Allocate_Memory Routine'
;    0790  1
;    0791  1 GLOBAL ROUTINE MEN$Allocate_Memory (R_SIZE, I_ADDRESS, I_RETSIZE) =
;    0792  1
;    0793  1 !++
;    0794  1 ! FUNCTIONAL DESCRIPTION:
;    0795  1 !
;    0796  1 !    Allocates specified amount of memory from VDS Non-Paged Pool and
;    0797  1 !    initializes the space with zero's.
;    0798  1 !
;    0799  1 ! INPUT PARAMETERS
;    0800  1 !
;    0801  1 !    R_SIZE  Address location containing size in bytes of memory space requested
;    0802  1 !
;    0803  1 ! OUTPUT PARAMETERS
;    0804  1 !
;    0805  1 !    I_ADDRESS  Address location to return address pointer to start
;    0806  1 !    of allocated memory
;    0807  1 !    I_RETSIZE  Address location to return size in bytes of allocated memory
;    0808  1 !
;    0809  1 ! IMPLICIT INPUTS
;    0810  1 !
;    0811  1 ! EXPLICIT OUTPUTS
;    0812  1 !
;    0813  1 ! SIDE EFFECTS
;    0814  1 !
;    0815  1 !    On error a message is printed
;    0816  1 !
;    0817  1 ! COMPLETION CODES
;    0818  1 !
;    0819  1 !    CRD$K_Success
;    0820  1 !    CRD$K_Failure
;    0821  1 !--
;    0822  2 BEGIN
;    0823  2     LOCAL
;    0824  2  Returned_Size,    ! The size returned by the allocation routine
;    0825  2  Returned_Address;  ! The address returned by the allocation routine
;    0826  2
;    0827  2     BIND ROUTINE
;    0828  2  CRD$Allocate_Memory = (.CRD$Exe$AloNonPaged) : JSB_Alloc;
;    0829  2     ! Get past the indirection
;    0830  2
;    0831  2     BIND
;    0832  2  Size = .R_SIZE,
```

```
;    0833  2  Address = .I_ADDRESS,
;    0834  2     RetSize = .I_RETSIZE;
;    0835  2
;    0836  3     IF (NOT (CRD$Allocate_Memory (.Size; Returned_Size, Returned_Address)))
;    0837  2     THEN
;    0838  3  BEGIN
;    0839  3  $CRD_Message (New_Line, MEN$GT_NoAll_NonPgMem);
;    0840  3  RETURN (CRD$K_Failure);
;    0841  3  END
;    0842  2     ELSE
;    0843  3  BEGIN
;    0844  3  CH$FILL (0,.Returned_Size, .Returned_Address);
;    0845  3   Address = .Returned_Address;
;    0846  3  RetSize = .Returned_Size;
;    0847  2  END;
;    0848  2
;    0849  2     IF (.CRD$GB_CRD_Debug) THEN
; P  0850  2  $CRD_Print ($ASCIC ('MEN$Allocate_Memory: Address: !XL(X), Size: !XL(X)!/'),
;    0851  2   .Returned_Address, .Returned_Size);
;    0852  2
;    0853  2     RETURN (CRD$K_Success);
;    0854  1 END;



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

4D  5F  65  74  61  63  6F  6C  6C  41  24  4E  45  4D  34  0015E P.AAJ:  .ASCII  \4MEN$Allocate_Memory: Address: !XL(X), S\  ;
3A  73  73  65  72  64  64  41  20  3A  79  72  6F  6D  65  0016D                ;
      53  20  2C  29  58  28  4C  58  21  20  0017C                ;
 2F  21  29  58  28  4C  58  21  20  3A  65  7A  69  00186    .ASCII  \ize: !XL(X)!/\         ;

            .EXTRN  CRD$EXE$ALONONPAGED
            .EXTRN  MEN$GT_NOALL_NONPGMEM
            .EXTRN  CRD$GT_NEW_LINE_MESSAGE

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0FFC 00000     .ENTRY  MEN$ALLOCATE_MEMORY, Save R2,R3,R4,R5,R6,-  ; 0791
             R7,R8,R9,R10,R11          ;
            51 04    BC  D0 00002     MOVL    aR_SIZE, R1           ; 0836
       00000000G  FF  16 00006     JSB     aCRD$EXE$ALONONPAGED          ;
            57       51  D0 0000C     MOVL    R1, R7           ;
            56       52  D0 0000F     MOVL    R2, R6           ;
            24       50  E8 00012     BLBS    R0, 1$           ;
       00000000G  EF  9F 00015     PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 0839
            01  DD 0001B     PUSHL   #1               ;
            1A  DD 0001D     PUSHL   #26              ;
       00000000G  FF       03  FB 0001F     CALLS   #3, aCRD$PRINT          ;
       00000000G  EF  9F 00026     PUSHAB  MEN$GT_NOALL_NONPGMEM          ;
            01  DD 0002C     PUSHL   #1               ;
            1A  DD 0002E     PUSHL   #26              ;
       00000000G  FF       03  FB 00030     CALLS   #3, aCRD$PRINT          ;
            2D  11 00037     BRB     3$               ; 0840
    57          00         6E       00  2C 00039 1$:   MOVC5   #0, (SP), #0, RETURNED_SIZE, -       ; 0844
```

```
                66     0003E     (RETURNED_ADDRESS)          ;
          08    BC      56  D0 0003F     MOVL     RETURNED_ADDRESS, @I_ADDRESS        ; 0845
          0C    BC      57  D0 00043     MOVL     RETURNED_SIZE, @I_RETSIZE           ; 0846
                14 00000000G  EF  E9 00047     BLBC     CRD$GB_CRD_DEBUG, 2$          ; 0849
                7E      56  7D 0004E     MOVQ     RETURNED_ADDRESS, -(SP)           ; 0851
           00000000'  EF  9F 00051     PUSHAB   P.AAJ               ;
                01  DD 00057     PUSHL    #1              ;
                1A  DD 00059     PUSHL    #26             ;
           00000000G  FF      05  FB 0005B     CALLS    #5, @CRD$PRINT            ;
                50      01  D0 00062 2$:    MOVL     #1, R0            ; 0853
            04 00065     RET                 ;
                50  D4 00066 3$:    CLRL     R0              ; 0854
            04 00068     RET                 ;

; Routine Size:  105 bytes,    Routine Base:  CODE + 0441


;    0855  1 %SBTTL 'MEN$Load_File Routine'
;    0856  1
;    0857  1 GLOBAL ROUTINE MEN$Load_File (R_FILE, R_FILE_DEFAULT, I_SIZE, I_ADDRESS, I_MEMSIZE)=
;    0858  1
;    0859  1 !++
;    0860  1 ! FUNCTIONAL DESCRIPTION:
;    0861  1 !
;    0862  1 !    Loads specified file image into memory.
;    0863  1 !
;    0864  1 ! INPUT PARAMETERS
;    0865  1 !
;    0866  1 !    R_FILE  Address of File Name ASCID
;    0867  1 !    R_FILE_DEFAULT Address of default File name ASCID
;    0868  1 !
;    0869  1 ! OUTPUT PARAMETERS
;    0870  1 !
;    0871  1 !    I_SIZE  Address location to return size in bytes of
;    0872  1 !   loaded file
;    0873  1 !    I_ADDRESS   Address location to receive address pointer to
;    0874  1 !   start of loaded file
;    0875  1 !    I_MEMSIZE  Address location to return size in bytes of allocated
;    0876  1 !   memory space for file
;    0877  1 !
;    0878  1 ! IMPLICIT INPUTS
;    0879  1 !
;    0880  1 ! EXPLICIT OUTPUTS
;    0881  1 !
;    0882  1 ! SIDE EFFECTS
;    0883  1 !
;    0884  1 !    If file load error a message is printed.
;    0885  1 !
;    0886  1 ! COMPLETION CODES
;    0887  1 !
;    0888  1 !    CRD$K_Success File loaded successfully
;    0889  1 !    CRD$K_Failure Error in file load
;    0890  1 !--
```

```
;    0891  1
;    0892  2 BEGIN
;    0893  2     LOCAL
;    0894  2  Status,
;    0895  2  File_Attributes,
;    0896  2  fab : $fab (),     ! FAB for file access [04]
;    0897  2  xab : $xabfhc ();   ! XAB for file access [04]
;    0898  2
;    0899  2
; P  0900  2 $fab_init (fab = fab, fna = .(.r_file+4), fns = .(.r_file),
; P  0901  2     dna = .(.r_file_default+4), dns = .(.r_file_default),
;    0902  2     xab = xab, fac = <get,bio>);  ![04]
;    0903  2
;    0904  2 if not (status = $open (fab = fab)) then Load_Error_Exit (.status); ![04]
;    0905  2
;    0906  2 $close (fab = fab);       ![04]
;    0907  2
;    0908  2 .I_Size = (.xab [xab$l_ebk]^9 + .xab [xab$w_ffb] - 512);  ![04]
;    0909  2
;    0910  2   file_attributes <0, 16, 0> = .fab [fab$w_mrs];   ![04]
;    0911  2   file_attributes <16, 8, 0> = .fab [fab$b_rfm];
;    0912  2   file_attributes <24, 8, 0> = .fab [fab$b_fsz];
;    0913  2
;    0914  2
;    0915  3     IF (NOT MEN$Allocate_Memory (.I_Size, .I_Address, .I_MemSize)) ![04]
;    0916  2     THEN
;    0917  2  Internal_Error_Exit (CRD$K_Allocation_Error, .(.I_Size), .(.I_MemSize), .(.I_Address));
;    0918  2
; P  0919  2     Status = $DS_LOAD (
; P  0920  2    File =      .R_file,
; P  0921  2    Default =   .R_file_Default,
; P  0922  2    Length =    .(.I_Size),
; P  0923  2    Address =   .(.I_Address),
; P  0924  2    RetLen =    .I_Size,
; P  0925  2    RetRec =    File_Attributes
;    0926  2    );
;    0927  3     IF (NOT .Status)
;    0928  2     THEN
;    0929  2  Load_Error_Exit (.Status);
;    0930  2
;    0931  2     RETURN (CRD$K_Success);
;    0932  1 END;


          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

          00193    .BLKB   1
      03  00194 P.AAK:  .BYTE   3          ;
      50  00195    .BYTE   80          ;
           0000  00196    .WORD   0          ;
       00000000  00198    .LONG   0          ;
       00000000  0019C    .LONG   0          ;
       00000000  001A0    .LONG   0          ;
       00000000  001A4    .LONG   0          ;
```

```
                0000  001A8     .WORD   0               ;
          02    001AA     .BYTE   2               ;
          00    001AB     .BYTE   0               ;
           00000000  001AC     .LONG   0               ;
          00    001B0     .BYTE   0               ;
          00    001B1     .BYTE   0               ;
          00    001B2     .BYTE   0               ;
          02    001B3     .BYTE   2               ;
           00000000  001B4     .LONG   0               ;
           00000000  001B8     .LONG   0               ;
           00000000  001BC     .LONG   0               ;
           00000000  001C0     .LONG   0               ;
           00000000  001C4     .LONG   0               ;
          00    001C8     .BYTE   0               ;
          00    001C9     .BYTE   0               ;
                0000  001CA     .WORD   0               ;
           00000000  001CC     .LONG   0               ;
                0000  001D0     .WORD   0               ;
          00    001D2     .BYTE   0               ;
          00    001D3     .BYTE   0               ;
           00000000  001D4     .LONG   0               ;
           00000000  001D8     .LONG   0               ;
                0000  001DC     .WORD   0               ;
          00    001DE     .BYTE   0               ;
          00    001DF     .BYTE   0               ;
           00000000  001E0     .LONG   0               ;
          1D    001E4 P.AAL:  .BYTE   29              ;
          2C    001E5     .BYTE   44              ;
                0000  001E6     .WORD   0               ;
           00000000  001E8     .LONG   0               ;
           00000000# 001EC     .LONG   0[9]            ;

        .EXTRN  SYS$OPEN, SYS$CLOSE
        .EXTRN  DS$LOAD

        .PSECT  CODE,NOWRT,  SHR,  PIC,2

          003C 00000     .ENTRY  MEN$LOAD_FILE, Save R2,R3,R4,R5       ; 0857
        5E 80    AE  9E 00002     MOVAB   -128(SP), SP            ;
    30  AE 00000000'  EF      0050   8F  28 00006     MOVC3   #80, P.AAK, FAB         ; 0896
    04  AE 00000000'  EF       2C  28 00011     MOVC3   #44, P.AAL, XAB         ; 0897
0050    8F           00         6E      00  2C 0001A     MOVC5   #0, (SP), #0, #80, $RMS_PTR         ; 0902
    30  AE      00021           ;
    30  AE      5003   8F  B0 00023     MOVW    #20483, $RMS_PTR            ;
    46  AE       22  90 00029     MOVB    #34, $RMS_PTR+22            ;
    4F  AE       02  90 0002D     MOVB    #2, $RMS_PTR+31             ;
    54  AE 04   AE  9E 00031     MOVAB   XAB, $RMS_PTR+36            ;
        55 04   AC  D0 00036     MOVL    R_FILE, R5              ;
    5C  AE 04   A5  D0 0003A     MOVL    4(R5), $RMS_PTR+44          ;
        54 08   AC  D0 0003F     MOVL    R_FILE_DEFAULT, R4          ;
    60  AE 04   A4  D0 00043     MOVL    4(R4), $RMS_PTR+48          ;
    64  AE       65  90 00048     MOVB    (R5), $RMS_PTR+52           ;
    65  AE       64  90 0004C     MOVB    (R4), $RMS_PTR+53           ;
    30  AE  9F 00050     PUSHAB  FAB             ; 0904
     00000000G  00       01  FB 00053     CALLS   #1, SYS$OPEN            ;
```

```
                53      50  D0 0005A     MOVL    R0, STATUS          ;
                16      53  E8 0005D     BLBS    STATUS, 3$          ;
            00018292    8F      53  D1 00060     CMPL    STATUS, #98962          ;
                04  12 00067     BNEQ    1$              ;
                0F  DD 00069     PUSHL   #15             ;
                02  11 0006B     BRB     2$              ;
                10  DD 0006D 1$:   PUSHL   #16             ;
            00000000G   EF      01  FB 0006F 2$:   CALLS   #1, CRD$STOP            ;
           30   AE  9F 00076 3$:   PUSHAB  FAB             ; 0906
            00000000G   00      01  FB 00079     CALLS   #1, SYS$CLOSE           ;
                52 0C   AC  D0 00080     MOVL    I_SIZE, R2          ; 0908
                50   14   AE      09  78 00084     ASHL    #9, XAB+16, R0          ;
                51 18   AE  3C 00089     MOVZWL  XAB+20, R1          ;
                62      FE00 C140   9E 0008D     MOVAB   -512(R1)[R0], (R2)          ;
                6E 66   AE  B0 00093     MOVW    FAB+54, FILE_ATTRIBUTES         ; 0910
           02   AE 4F   AE  90 00097     MOVB    FAB+31, FILE_ATTRIBUTES+2           ; 0911
           03   AE 6F   AE  90 0009C     MOVB    FAB+63, FILE_ATTRIBUTES+3           ; 0912
                7E 10   AC  7D 000A1     MOVQ    I_ADDRESS, -(SP)            ; 0915
                52  DD 000A5     PUSHL   R2              ;
           FEEB CF      03  FB 000A7     CALLS   #3, MEN$ALLOCATE_MEMORY         ;
                1B      50  E8 000AC     BLBS    R0, 4$              ;
             10 BC  DD 000AF     PUSHL   aI_ADDRESS          ; 0917
             14 BC  DD 000B2     PUSHL   aI_MEMSIZE          ;
                62  DD 000B5     PUSHL   (R2)                ;
                7E      01  CE 000B7     MNEGL   #1, -(SP)           ;
            00000000G   EF      04  FB 000BA     CALLS   #4, MEN$MENU_TEST_INTERNAL_ERROR        ;
                0D  DD 000C1     PUSHL   #13             ;
            00000000G   EF      01  FB 000C3     CALLS   #1, CRD$STOP            ;
                01  DD 000CA 4$:   PUSHL   #1              ; 0926
             04 AE  9F 000CC     PUSHAB  FILE_ATTRIBUTES         ;
                52  DD 000CF     PUSHL   R2              ;
             10 BC  DD 000D1     PUSHL   aI_ADDRESS          ;
                62  DD 000D4     PUSHL   (R2)                ;
                54  DD 000D6     PUSHL   R4              ;
                55  DD 000D8     PUSHL   R5              ;
            00000000G   9F      07  FB 000DA     CALLS   #7, a#DS$LOAD           ;
                53      50  D0 000E1     MOVL    R0, STATUS          ;
                16      53  E8 000E4     BLBS    STATUS, 7$          ; 0927
            00018292    8F      53  D1 000E7     CMPL    STATUS, #98962          ; 0929
                04  12 000EE     BNEQ    5$              ;
                0F  DD 000F0     PUSHL   #15             ;
                02  11 000F2     BRB     6$              ;
                10  DD 000F4 5$:   PUSHL   #16             ;
            00000000G   EF      01  FB 000F6 6$:   CALLS   #1, CRD$STOP            ;
                50      01  D0 000FD 7$:   MOVL    #1, R0          ; 0931
             04 00100    RET                 ; 0932

; Routine Size:  257 bytes,    Routine Base:  CODE + 04AA


;   0933  1 %SBTTL 'MEN$Find_SupBlk Routine'
;   0934  1
;   0935  1 GLOBAL ROUTINE MEN$Find_SupBlk (R_DATABASE, I_DEVNAM, I_SUPPORT_BLK) -
;   0936  1
;   0937  1 !++
```

```
;    0938  1 ! FUNCTIONAL DESCRIPTION:
;    0939  1 !
;    0940  1 !    Searches for a support block denoted by a  device name in the
;    0941  1 !    database memory area specified.  The support block must be valid
;    0942  1 !    (SupBlk$V_Status_Invalid = False).
;    0943  1 !
;    0944  1 ! INPUT PARAMETERS
;    0945  1 !
;    0946  1 !    R_DATABASE  Address of 2-longword array point ng to database
;    0947  1 !        [0] = Size in bytes of database
;    0948  1 !       [1] = Address pointer to start of database
;    0949  1 !    I_DEVNAME  Address of device name ASCIC
;    0950  1 !
;    0951  1 ! OUTPUT PARAMETERS
;    0952  1 !
;    0953  1 !    I_SUPPORT_BLK Address to receive the address pointer of the
;    0954  1 !   support block
;    0955  1 !
;    0956  1 ! COMPLETION CODES
;    0957  1 !
;    0958  1 ! CRD$K_Success  Support Block Found
;    0959  1 ! CRD$K_Failure  Support Block Not Found.  The support block
;    0960  1 !    for the device may not exist or it may exist
;    0961  1 !    but is considered invalid if the SupBlk$V_Status_Invalid
;    0962  1 !    bit is tru..
;    0963  1 !--
;    0964  2 BEGIN
;    0965  2     LOCAL
;    0966  2  SupBlk_Ptr :  REF $MEN Supblk_ATTR,
;    0967  2  SupBlk_Found,
;    0968  2  End_of_Database;
;    0969  2
;    0970  2     BIND
;    0971  2  Database    = .R_DATABASE  : VECTOR [2, LONG],
;    0972  2  Devnam      = .I_Devnam  : VECTOR [, BYTE],
;    0973  2  Support_Blk = .I_Support_Blk;
;    0974  2
;    0975  2     SupBlk_Found    = False;
;    0976  2     End_of_DataBase = .DataBase [0] + .DataBase [1];
;    0977  2     SupBlk_Ptr      = .DataBase [1];
```

```
;    0978  2     WHILE ((.SupBlk_Ptr LSS .End_of_DataBase) AND (NOT .SupBlk_Found)) DO
;    0979  3  BEGIN
;    0980  3  IF  CH$EQL
;    0981  3 !    .    .   Compare device strings
;    0982  3    (
;    0983  3    .Devnam [0],
;    0984  3    Devnam [1],
;    0985  3    .(SupBlk_Ptr [SupBlk$T_Device_Name])<0,8>,
;    0986  3    (SupBlk_Ptr [SupBlk$T_Device_Name]) + 1,
;    0987  3    %C' '
;    0988  3    )
;    0989  3       AND
;    0990  3 !     .    .    Check for validity of block
;    0991  4    (NOT .SupBlk_Ptr [SupBlk$V_Status_Invalid])
;    0992  3            AND
;    0993  3 !    .     .    Check that in correct mode, on/off line, and correct diag level
;    0994  4    (
;    0995  5      (
;    0996  6      (.SupBlk_Ptr[SupBlk$W_Level] EQL '2 ')     ! [03]
;    0997  5      AND          ! [03]
;    0998  6      ( DSA$V_CRD_Menutest_ON)       ! [03]
;    0999  5      )
;    1000  4    OR           ! [03]
;    1001  5      (
;    1002  6      (.SupBlk_Ptr[SupBlk$W_Level] EQL '2R')     ! [03]
;    1003  5      AND          ! [03]
;    1004  6      (.DSA$V_CRD_Menutest ON)       ! [03]
;    1005  5      )
;    1006  4    OR           ! [03]
;    1007  5      (
;    1008  6      (.SupBlk_Ptr[SupB k$W_Level] EQL '3 ')     ! [03]
;    1009  5      AND          ! [03]
;    1010  6      (.DSA$V_CRD_Menutest_Off)       ! [03]
;    1011  5      )
;    1012  4     )
;    1013  3  THEN
;    1014  3        SupBlk_Found = True
;    1015  3  ELSE
;    1016  3        SupBlk_Ptr = (.SupBlk_Ptr [SupBlk$L_SupBlk_Size] * 4)  + (.SupBlk_Ptr + 4);
;    1017  3        ! Point to next Support Block
;    1018  2  END;
;    1019  2
;    1020  3     IF (NOT .SupBlk_Found)
;    1021  2     THEN
;    1022  3  RETURN (CRD$K_Failure)
;    1023  2     ELSE
;    1024  2  Support_Blk = .SupBlk_Ptr;
;    1025  2
;    1026  2     RETURN (CRD$K_Success);
;    1027  1 END;
```

```
                00FC 00000     .ENTRY  MEN$FIND_SUPBLK, Save R2,R3,R4,R5,R6,R7     ; 0935
              50 04   AC  D0 00002     MOVL    R_DATABASE, R0                      ; 0971
              55 08   AC  D0 00006     MOVL    I_DEVNAM, R5                        ; 0972
              56  D4 0000A     CLRL    SUPBLK_FOUND                        ; 0975
              57       60 04   A0  C1 0000C     ADDL3   4(R0), (R0), END_OF_DATABASE        ; 0976
              54 04   A0  D0 00011     MOVL    4(R0), SUPBLK_PTR                   ; 0977
              51  11 00015     BRB     6$                          ; 0978
              51       65  9A 00017 1$:   MOVZBL  (R5), R1                    ; 0983
              50 04   A4  9A 0001A     MOVZBL  4(SUPBLK_PTR), R0                   ; 0985
    50            20   01   A5       51  2D 0001E     CMPC5   R1, 1(R5), #32, R0, 5(SUPBLK_PTR)       ; 0984
      05    A4      00024                ;
              38  12 00026     BNEQ    5$                          ;
              34 13   A4  E8 00028     BLBS    19(SUPBLK_PTR), 5$                  ; 0991
    2032      8F 15   A4  B1 0002C     CMPW    21(SUPBLK_PTR), #8242               ; 0996
              08  12 00032     BNEQ    2$                          ;
              1F 0000FE02   9F       02  E0 00034     BBS     #2, @#^X0000FE02, 4$                ; 0998
    5232      8F 15   A4  B1 0003C 2$:   CMPW    21(SUPBLK_PTR), #21042              ; 1002
              08  12 00042     BNEQ    3$                          ;
              0F 0000FE02   9F       02  E0 00044     BBS     #2, @#^X0000FE02, 4$                ; 1004
    2033      8F 15   A4  B1 0004C 3$:   CMPW    21(SUPBLK_PTR), #8243               ; 1008
              0C  12 00052     BNEQ    5$                          ;
              05 0000FE02   9F  E9 00054     BLBC    @#^X0000FE02, 5$                    ; 1010
              56       01  D0 0005B 4$:   MOVL    #1, SUPBLK_FOUND                    ; 1014
              08  11 0005E     BRB     6$                          ;
              50       64  D0 00060 5$:   MOVL    (SUPBLK_PTR), R0                    ; 1016
              54 04 A440  DE 00063     MOVAL   4(SUPBLK_PTR)[R0], SUPBLK_PTR       ;
              57       54  D1 00068 6$:   CMPL    SUPBLK_PTR, END_OF_DATABASE         ; 0978
              03  18 0006B     BGEQ    7$                          ;
              A7       56  E9 0006D     BLBC    SUPBLK_FOUND, 1$            ;
              08       56  E9 00070 7$:   BLBC    SUPBLK_FOUND, 8$                    ; 1020
        0C    BC       54  D0 00073     MOVL    SUPBLK_PTR, @I_SUPPORT_BLK          ; 1024
              50       01  D0 00077     MOVL    #1, R0                      ; 1026
          04 0007A     RET                         ;
              50  D4 0007B 8$:   CLRL    R0                  ; 1027
          04 0007D     RET                         ;

; Routine Size:  126 bytes,    Routine Base:  CODE + 05AB


;   1028  1 %SBTTL 'MEN$Dump_TstQ_HST Routine'
;   1029  1
;   1030  1 GLOBAL ROUTINE MEN$Dump_TstQ_HST =
;   1031  1
;   1032  1 !++
;   1033  1 ! FUNCTIONAL DESCRIPTION:
;   1034  1 !
;   1035  1 !    Prints the 'test start text' and hardware support table
;   1036  1 !    info associated with each of the Test Queue Table entries.
;   1037  1 !
;   1038  1 ! INPUT PARAMETERS
;   1039  1 !
;   1040  1 ! OUTPUT PARAMETERS
;   1041  1 !
;   1042  1 ! IMPLICIT INPUTS
;   1043  1 !
```

```
;    1044  1 !    Test Queue pointed to by MEN$GA_Test_Queue
;    1045  1 !
;    1046  1 ! EXPLICIT OUTPUTS
;    1047  1 !
;    1048  1 ! SIDE EFFECTS
;    1049  1 !
;    1050  1 ! COMPLETION CODES
;    1051  1 !
;    1052  1 !    CRD$K_Success;
;    1053  1 !--
;    1054  2 BEGIN
;    1055  2     LOCAL
;    1056  2  Diagnostic_Name_Ptr,
;    1057  2  Set_Flags_Args_Ptr,
;    1058  2  Set_Event_Args_Ptr,
;    1059  2  Device_Name_Ptr,
;    1060  2  Start_Qualifiers_Ptr,
;    1061  2  Test_Start_Text_Ptr,
;    1062  2  RunTime_Ptr,
;    1063  2  Preparation_Error_Text_Ptr,
;    1064  2  TstQ_Tbl_Index,
;    1065  2  TstQ_Tbl_Entries,
;    1066  2  TstQ_Tbl_Ptr : REF VECTOR [, LONG],
;    1067  2  SupBlk_Ptr : REF $MEN_SupBlk_ATTR,
;    1068  2  DDB_Ptr  : REF Device_Data_Block_Structure,
;    1069  2  PTable_Ptr : REF $DS_HPO_DECL (),
;    1070  2  HST_No,
;    1071  2  HST_Index,
;    1072  2  HST_Ptr  : REF VECTOR [CRD$K_Last_Entry, LONG];
;    1073  2
;    1074  2     BIND
;    1075  2  T_HST_Entry_Null =
;    1076  2      $ASCIC (''),
;    1077  2
;    1078  2  T_HST_Title =
;    1079  2      $ASCIC ('--- Hardware Support Table Information ---'),
;    1080  2
;    1081  2  T_Dump_HST =
; P  1082  2       $ASCIC ('!/',
; P  1083  2   'Diagnostic: .!AC.!/',
; P  1084  2   'Flags:      .!AC.!/',
; P  1085  2   'Events:     .!AC.!/',
; P  1086  2   'Device:     .!AC.!/',
; P  1087  2   'Start:      .!AC.!/',
; P  1088  2   'Start Text: .!AC.!/',
; P  1089  2   'Time:       .!AC.!/',
;    1090  2   'Prep Text:  .!AC.!/');
;    1091  2
;    1092  2     !+
;    1093  2     ! Set up for indexing into Test Queue Table
;    1094  2     !-
;    1095  2
;    1096  2     TstQ_Tbl_Index = 0;
;    1097  2     TstQ_Tbl_Entries = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb Entries];
;    1098  2     TstQ_Tbl_Ptr = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr];
```

```
;    1099  2
;    1100  2      !+
;    1101  2      ! Get a DDB pointer from the Test Queue Table until first null pointer
;    1102  2      !-
;    1103  2
;    1104  2      WHILE (.TstQ_Tbl_Index LSS .TstQ_Tbl_Entries ) AND
;    1105  2        ( (DDB_Ptr =.TstQ_Tbl_Ptr [.TstQ_Tbl_Index]) NEQ 0) DO
;    1106  3    BEGIN
;    1107  3    IF ( HST_Ptr = .DDB_Ptr [CRD$K_DDB_Menu_Support_Entry]) NEQ 0
;    1108  3    THEN
;    1109  4        BEGIN
;    1110  4        HST_No = .DDB_Ptr [CRD$K_DDB_Numb_Support_Tables];
;    1111  4        HST_Index = - 1;
;    1112  4
;    1113  4        WHILE ( (HST_Index = .HST_Index + 1) LSS .HST_No ) DO
;    1114  5         BEGIN
;    1115  5     !+
;    1116  5     ! Print the 'test start text' associated with the HST
;    1117  5     !-
;    1118  5
;    1119  5     MEN$FncTst_TestStrt_Text (.DDB_Ptr, .HST_Ptr);
;    1120  5
;    1121  5     !+
;    1122  5     ! Print the HST title
;    1123  5     !-
;    1124  5
;    1125  5       $CRD_Message (New_Line, T_HST_Title);
;    1126  5
;    1127  5     !+
;    1128  5     ! Check for null ASCIC pointers
;    1129  5     !-
;    1130  5
;    1131  5     IF (.HST_Ptr [CRD$K_Diagnostic_Name]) EQL 0
;    1132  5     THEN
;    1133  6         (Diagnostic_Name_Ptr = T_HST_Entry_Null)
;    1134  5     ELSE
;    1135  5         Diagnostic_Name_Ptr = .HST_Ptr [CRD$K_Diagnostic_Name];
;    1136  5
;    1137  5     IF (.HST_Ptr [CRD$K_Set_Flags Args]) EQL 0
;    1138  5     THEN
;    1139  6         (Set_Flags_Args_Ptr = T_HST_Entry_Null)
;    1140  5     ELSE
;    1141  5         Set_Flags_Args_Ptr = .HST_Ptr [CRD$K_Set_Flags_Args];
;    1142  5
;    1143  5     IF (.HST_Ptr [CRD$K_Set_Event_Args]) EQL 0
;    1144  5     THEN
;    1145  6         (Set_Event_Args_Ptr = T_HST_Entry_Null)
;    1146  5     ELSE
;    1147  5         Set_Event_Args_Ptr = .HST_Ptr [CRD$K_Set_Event_Args];
;    1148  5
;    1149  5     IF (.HST_Ptr [CRD$K_Device_Name]) EQL 0
;    1150  5     THEN
;    1151  6         (Device_Name_Ptr = T_HST_Entry_Null)
;    1152  5     ELSE
;    1153  5         Device_Name_Ptr = .HST_Ptr [CRD$K_Device_Name];
```

```
;    1154  5
;    1155  5    IF (.HST_Ptr [CRD$K_Start_Qualifiers]) EQL 0
;    1156  5    THEN
;    1157  6        (Start_Qualifiers_Ptr = T_HST_Entry_Null)
;    1158  5    ELSE
;    1159  5        Start_Qualifiers_Ptr = .HST_Ptr [CRD$K_Start_Qualifiers];
;    1160  5
;    1161  5    IF (.HST_Ptr [CRD$K_Test_Start_Text]) EQL 0
;    1162  5    THEN
;    1163  6        (Test_Start_Text_Ptr = T_HST_Entry_Null)
;    1164  5    ELSE
;    1165  5        Test_Start_Text_Ptr = .HST_Ptr [CRD$K_Test_Start_Text];
;    1166  5
;    1167  5    IF (.HST_Ptr [CRD$K_RunTime]) EQL 0
;    1168  5    THEN
;    1169  6        (RunTime_Ptr = T_HST_Entry_Null)
;    1170  5    ELSE
;    1171  5        RunTime_Ptr = .HST_Ptr [CRD$K_RunTime];
;    1172  5
;    1173  5    IF (.HST_Ptr [CRD$K_Preparation_Error_Text]) EQL 0
;    1174  5    THEN
;    1175  6        (Preparation_Error_Text_Ptr = T_HST_Entry_Null)
;    1176  5    ELSE
;    1177  5        Preparation_Error_Text_Ptr = .HST_Ptr [CRD$K_Preparation_Error_Text];
;    1178  5
;    1179  5    !+
;    1180  5    ! Print the HST info
;    1181  5    !-
;    1182  5
; P  1183  5        $CRD_Message (New_Line, T_Dump_HST,
; P  1184  5      .Diagnostic_Name_Ptr,
; P  1185  5      .Set_Flags_Args_Ptr,
; P  1186  5      .Set_Event_Args_Ptr,
; P  1187  5      .Device_Name_Ptr,
; P  1188  5      .Start_Qualifiers_Ptr,
; P  1189  5      .Test_Start_Text_Ptr,
; P  1190  5      .RunTime_Ptr,
; P  1191  5      .Preparation_Error_Text_Ptr
;    1192  5      );
;    1193  5
;    1194  5     HST_Ptr = .Hst_Ptr + (CRD$K_Last_Entry * 4);
;    1195  5       ! Point to the next HST
;    1196  4    END;
;    1197  3       END;    ! HST Exists
;    1198  3
;    1199  3   TstQ_Tbl_Index = .TstQ_Tbl_Index +1;
;    1200  3       ! Next entry
;    1201  2   END;     ! All Test Queue Table Entries
;    1202  2      RETURN (CRD$K_Success);
;    1203  1 END;


            .PSECT  DATA,NOWRT,NOEXE,  SHR,2
```

```
                00  00210 P.AAM:  .ASCII  <0>                ;
53  20  65  72  61  77  64  72  61  48  20  2D  2D  2D  2A  00211 P.AAN:  .ASCII  \*--- Hardware Support Table Information \  ;
6E  49  20  65  6C  62  61  54  20  74  72  6F  70  70  75  00220                ;
              20  6E  6F  69  74  61  6D  72  6F  66  0022F                    ;
              2D  2D  2D  00239     .ASCII  \---\                ;
20  3A  63  69  74  73  6F  6E  67  61  69  44  2F  21  9A  0023C P.AAO:  .ASCII  <154>\!/Diagnostic: .!AC.!/Flags:      .\   ;
20  20  3A  73  67  61  6C  46  2F  21  2E  43  41  21  2E  0024B                ;
          2E  20  20  20  20  0025A                    ;
20  20  3A  73  74  6E  65  76  45  2F  21  2E  43  41  21  0025F     .ASCII  \!AC.!/Events:     .!AC.!/Device:     .!A\  ;
63  69  76  65  44  2F  21  2E  43  41  21  2E  20  20  20  0026E                ;
          41  21  2E  20  20  20  20  20  3A  65  0027D                        ;
20  20  20  20  20  3A  74  72  61  74  53  2F  21  2E  43  00287     .ASCII  \C.!/Start:      .!AC.!/Start Text: ..AC.\  ;
54  20  74  72  61  74  53  2F  21  2E  43  41  21  2E  20  00296                ;
          2E  43  41  21  2E  20  3A  74  78  65  002A5                        ;
2E  20  20  20  20  20  20  20  3A  65  6D  69  54  2F  21  002AF     .ASCII  \!/Time:       .!AC.!/Prep Text:  .!AC.!/\  ;
74  78  65  54  20  70  65  72  50  2F  21  2E  43  41  21  002BE                ;
          2F  21  2E  43  41  21  2E  20  20  3A  002CD                        ;

        T_HST_ENTRY_NULL=   P.AAM
        T_HST_TITLE=        P.AAN
        T_DUMP_HST=         P.AAO
         .EXTRN  MEN$FNCTST_TESTSTRT_TEXT

         .PSECT  CODE,NOWRT,  SHR,  PIC,2

           0FFC 00000     .ENTRY  MEN$DUMP_TSTQ_HST, Save R2,R3,R4,R5,R6,R7,- ; 1030
          R8,R9,R10,R11           ;
         5E      14  C2 00002     SUBL2   #20, SP          ;
         54  D4 00005     CLRL    TSTQ_TBL_INDEX          ; 1096
10   AE 00000000'  EF  D0 00007     MOVL    MEN$GA_TEST_QUEUE, TSTQ_TBL_ENTRIES        ; 1097
0C   AE 00000000'  EF  D0 0000F     MOVL    MEN$GA_TEST_QUEUE+4, TSTQ_TBL_PTR          ; 1098
   0115  31 00017     BRW     22$             ; 1104
     52 10   A3  D0 0001A 1$:   MOVL    16(DDB_PTR), HST_PTR          ; 1107
     03  12 0001E     BNEQ    2$              ;
   010A  31 00020     BRW     21$             ;
     6E 18   A3  D0 00023 2$:   MOVL    24(DDB_PTR), HST_NO           ; 1110
04   AE      01  CE 00027     MNEGL   #1, HST_INDEX           ; 1111
   00F3  31 0002B     BRW     20$             ; 1113
     52  DD 0002E 3$:   PUSHL   HST_PTR           ; 1119
     53  DD 00030     PUSHL   DDB_PTR         ;
 00000000G  EF      02  FB 00032     CALLS   #2, MEN$FNCTST_TESTSTRT_TEXT        ;
 00000000G  EF  9F 00039     PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 1125
     01  DD 0003F     PUSHL   #1              ;
     1A  DD 00041     PUSHL   #26             ;
 00000000G  FF      03  FB 00043     CALLS   #3, @CRD$PRINT          ;
 00000000'  EF  9F 0004A     PUSHAB  T_HST_TITLE             ;
     01  DD 00050     PUSHL   #1              ;
     1A  DD 00052     PUSHL   #26             ;
 00000000G  FF      03  FB 00054     CALLS   #3, @CRD$PRINT          ;
     62  D5 0005B     TSTL    (HST_PTR)           ; 1131
     0A  12 0005D     BNEQ    4$              ;
08   AE 00000000'  EF  9E 0005F     MOVAB   T_HST_ENTRY_NULL, DIAGNOSTIC_NAME_PTR       ; 1133
     04  11 00067     BRB     5$              ;
08   AE      62  D0 00069 4$:   MOVL    (HST_PTR), DIAGNOSTIC_NAME_PTR      ; 1135
04   A2  D5 0006D 5$:   TSTL    4(HST_PTR)          ; 1137
```

```
                09  12 00070     BNEQ    6$              ;
                5B 00000000'  EF  9E 00072     MOVAB   T_HST_ENTRY_NULL, SET_FLAGS_ARGS_PTR        ; 1139
                04  11 00079     BRB     7$              ;
                5B 04    A2  D0 0007B 6$:    MOVL    4(HST_PTR), SET_FLAGS_ARGS_PTR        ; 1141
          08    A2  D5 0007F 7$:    TSTL    8(HST_PTR)              ; 1143
                09  12 00082     BNEQ    8$              ;
                5A 00000000'  EF  9E 00084     MOVAB   T_HST_ENTRY_NULL, SET_EVENT_ARGS_PTR        ; 1145
                04  11 0008B     BRB     9$              ;
                5A 08    A2  D0 0008D 8$:    MOVL    8(HST_PTR), SET_EVENT_ARGS_PTR        ; 1147
          0C    A2  D5 00091 9$:    TSTL    12(HST_PTR)             ; 1149
                09  12 00094     BNEQ    10$             ;
                59 00000000'  EF  9E 00096     MOVAB   T_HST_ENTRY_NULL, DEVICE_NAME_PTR         ; 1151
                04  11 0009D     BRB     11$             ;
                59 0C    A2  D0 0009F 10$:    MOVL    12(HST_PTR), DEVICE_NAME_PTR          ; 1153
          10    A2  D5 000A3 11$:    TSTL    16(HST_PTR)             ; 1155
                09  12 000A6     BNEQ    12$             ;
                58 00000000'  EF  9E 000A8     MOVAB   T_HST_ENTRY_NULL, START_QUALIFIERS_PTR        ; 1157
                04  11 000AF     BRB     13$             ;
                58 10    A2  D0 000B1 12$:    MOVL    16(HST_PTR), START_QUALIFIERS_PTR         ; 1159
          14    A2  D5 000B5 13$:    TSTL    20(HST_PTR)             ; 1161
                09  12 000B8     BNEQ    14$             ;
                57 00000000'  EF  9E 000BA     MOVAB   T_HST_ENTRY_NULL, TEST_START_TEXT_PTR         ; 1163
                04  11 000C1     BRB     15$             ;
                57 14    A2  D0 000C3 14$:    MOVL    20(HST_PTR), TEST_START_TEXT_PTR         ; 1165
          18    A2  D5 000C7 15$:    TSTL    24(HST_PTR)             ; 1167
                09  12 000CA     BNEQ    16$             ;
                56 00000000'  EF  9E 000CC     MOVAB   T_HST_ENTRY_NULL, RUNTIME_PTR         ; 1169
                04  11 000D3     BRB     17$             ;
                56 18    A2  D0 000D5 16$:    MOVL    24(HST_PTR), RUNTIME_PTR          ; 1171
          1C    A2  D5 000D9 17$:    TSTL    28(HST_PTR)             ; 1173
                09  12 000DC     BNEQ    18$             ;
                55 00000000'  EF  9E 000DE     MOVAB   T_HST_ENTRY_NULL, -           ; 1175
                 PREPARATION_ERROR_TEXT_PTR         ;
                04  11 000E5     BRB     19$             ;
                55 1C    A2  D0 000E7 18$:    MOVL    28(HST_PTR), PREPARATION_ERROR_TEXT_PTR       ; 1177
           00000000G  EF  9F 000EB 19$:    PUSHAB  CRD$GT_NEW_LINE_MESSAGE         ; 1192
                01  DD 000F1     PUSHL   #1              ;
                1A  DD 000F3     PUSHL   #26             ;
           00000000G  FF       03  FB 000F5     CALLS   #3, @CRD$PRINT          ;
                55  DD 000FC     PUSHL   PREPARATION_ERROR_TEXT_PTR          ;
                56  DD 000FE     PUSHL   RUNTIME_PTR             ;
                57  DD 00100     PUSHL   TEST_START_TEXT_PTR           ;
                58  DD 00102     PUSHL   START_QUALIFIERS_PTR            ;
                59  DD 00104     PUSHL   DEVICE_NAME_PTR           ;
                5A  DD 00106     PUSHL   SET_EVENT_ARGS_PTR          ;
                5B  DD 00108     PUSHL   SET_FLAGS_ARGS_PTR          ;
          24    AE  DD 0010A     PUSHL   DIAGNOSTIC_NAME_PTR           ;
           00000000'  EF  9F 0010D     PUSHAB  T_DUMP_HST              ;
                01  DD 00113     PUSHL   #1              ;
                1A  DD 00115     PUSHL   #26             ;
           00000000G  FF       0B  FB 00117     CALLS   #11, @CRD$PRINT          ;
                52       20  C0 0011E     ADDL2   #32, HST_PTR            ; 1194
          04    AE  D6 00121 20$:    INCL    HST_INDEX           ; 1113
                6E 04    AE  D1 00124     CMPL    HST_INDEX, HST_NO          ;
                03  18 00128     BGEQ    21$             ;
```

```
         FF01  31 0012A     BRW     3$                ;
           54  D6 0012D 21$:    INCL    TSTQ_TBL_INDEX          ; 1199
      10    AE       54  D1 0012F 22$:   CMPL    TSTQ_TBL_INDEX, TSTQ_TBL_ENTRIES        ; 1104
            0A  18 00133     BGEQ    23$               ;
            53 0C BE44  D0 00135     MOVL    @TSTQ_TBL_PTR[TSTQ_TBL_INDEX], DDB_PTR       ; 1105
            03  13 0013A     BEQL    23$               ;
         FEDB  31 0013C     BRW     1$                ;
           50       01  D0 0013F 23$:   MOVL    #1, R0          ; 1202
        04 00142     RET                ; 1203

; Routine Size:  323 bytes,    Routine Base:  CODE + 0629


;    1204  1 %SBTTL 'MEN$Menu_Cleanup Routine'
;    1205  1
;    1206  1 GLOBAL ROUTINE MEN$Menu_Cleanup =
;    1207  1
;    1208  1 !++
;    1209  1 ! FUNCTIONAL DESCRIPTION:
;    1210  1 !
;    1211  1 !    Performs cleanup tasks associated with CRD MENU TEST and specifically
;    1212  1 !    with respect to MEN$xxx code modules.
;    1213  1 !
;    1214  1 ! INPUT PARAMETERS
;    1215  1 !
;    1216  1 ! OUTPUT PARAMETERS
;    1217  1 !
;    1218  1 ! IMPLICIT INPUTS
;    1219  1 !
;    1220  1 !
;    1221  1 ! EXPLICIT OUTPUTS
;    1222  1 !
;    1223  1 ! SIDE EFFECTS
;    1224  1 !
;    1225  1 ! COMPLETION CODES
;    1226  1 !
;    1227  1 !    CRD$K_Success;
;    1228  1 !--
```

```
;    1229  1
;    1230  2 BEGIN
;    1231  2     LOCAL
;    1232  2  Next_DDB,
;    1233  2  DDB_Ptr  : REF Device_Data_Block_Structure,
;    1234  2  Queue_Head : REF $MEN_Queue_Link_Attr,
;    1235  2  Queue_BlkPtr : REF $MEN_Queue_Link_Attr;
;    1236  2
;    1237  2     !+
;    1238  2     ! Deallocate memory allocated for loaded support database
;    1239  2     !-
;    1240  2
;    1241  2     IF (.MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Ptr] NEQ 0) THEN
;    1242  3  BEGIN
;    1243  3  CRD$Dispose (.MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Total_Siz],
;    1244  3    .MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Start]);
;    1245  3
;    1246  3  MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Total_Siz] = 0;
;    1247  3  MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Ptr]  = 0;
;    1248  3  MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Size]  = 0;
;    1249  3  MEN$GA_FuncSup_DatBase [MEN$V_FuncSup_DatBase_Start]  = 0;
;    1250  2  END;
;    1251  2
;    1252  2     !+
;    1253  2     ! Deallocate memory allocated for Test Queue Table.
;    1254  2     !-
;    1255  2
;    1256  2     IF (.MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr] NEQ 0) THEN
;    1257  3  BEGIN
;    1258  3  CRD$Dispose (.MEN$GA_Test_Queue [MEN$V_TQ_Table_Size], .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr]);
;    1259  3
;    1260  3  MEN$GA_Test_Queue [MEN$V_TQ_Table_Size]  = 0;
;    1261  3  MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr]  = 0;
;    1262  3  MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb_Entries] = 0;
;    1263  2  END;
;    1264  2
;    1265  2     !+
;    1266  2     ! Deallocate the memory allocated for the HST's and the DDB's:
;    1267  2     ! Basic Algorithm:
;    1268  2     ! 1. Save the pointer to the next DDB
;    1269  2     ! 2. If there is one, dispose of the Hardware Support Block for this DDB.
;    1270  2     ! 3. Dispose of the DDB itself.
;    1271  2     ! 4. Go to the next DDB (saved in step 1).
;    1272  2     !-
;    1273  2
;    1274  2     DDB_Ptr = .MEN$GA_DDB_Head [QUEUE$L_FLink];
;    1275  2      ! Point to the first DDB
;    1276  2     IF (.DDB_Ptr NEQ 0) THEN  ! If the DDB's have been created, . . .
;    1277  3  BEGIN
;    1278  3  DO
;    1279  4   BEGIN
;    1280  4   Next_DDB = .DDB_Ptr [CRD$K_DDB_Flink];
;    1281  4
;    1282  4   IF (.DDB_Ptr [CRD$K_DDB_Menu_Support_Entry] NEQ 0) THEN
;    1283  4    CRD$Dispose (.DDB_Ptr [CRD$K_DDB_Menu_Support_Size], .DDB_Ptr [CRD$K_DDB_Menu_Support Entry]);
```

```
;    1284  4      ! Dispose of the Hardware Support Block, if there is one
;    1285  4
;    1286  4    CRD$Dispose (.DDB_Ptr [CRD$K_DDB_Memory_Size], .DDB_Ptr);
;    1287  4      ! Dispose of the DDB itself
;    1288  4
;    1289  4    DDB_Ptr = .Next_DDB; ! Advance to the next one in the queue
;    1290  4    END
;    1291  3   UNTIL (.DDB_Ptr EQL MEN$GA_DDB_Head);
;    1292  3      ! Loop until the last DDB's FLink points to the Queue Head
;    1293  3
;    1294  3   MEN$GA_DDB_Head [QUEUE$L_FLink] = 0;
;    1295  3   MEN$GA_DDB_Head [QUEUE$L_FLink] = 0;
;    1296  2   END;
;    1297  2
;    1298  2      !+
;    1299  2      ! Indicate that Initialization is now invalid
;    1300  2      !-
;    1301  2
;    1302  2      MEN$GB_FncTst_Init = False;
;    1303  2      RETURN (CRD$K_Success);
;    1304  1 END;


            .EXTRN  CRD$DISPOSE

                000C 00000     .ENTRY  MEN$MENU_CLEANUP, Save R2,R3        ; 1206
        00000000'  EF  D5 00002     TSTL    MEN$GA_FUNCSUP_DATBASE+4        ; 1241
            20  13 00008     BEQL    1$              ;
            7E 00000000'  EF  7D 0000A     MOVQ    MEN$GA_FUNCSUP_DATBASE+8, -(SP)       ; 1243
         00000000G  EF    02  FB 00011     CALLS   #2, CRD$DISPOSE         ;
         00000000'  EF  7C 00018     CLRQ    MEN$GA_FUNCSUP_DATBASE+4        ; 1247
         00000000'  EF  D4 0001E     CLRL    MEN$GA_FUNCSUP_DATBASE          ; 1248
         00000000'  EF  D4 00024     CLRL    MEN$GA_FUNCSUP_DATBASE+12       ; 1249
            50 00000000'  EF  D0 0002A 1$:   MOVL    MEN$GA_TEST_QUEUE+4, R0        ; 1256
            1B  13 00031     BEQL    2$              ;
            50  DD 00033     PUSHL   R0              ; 1258
         00000000'  EF  DD 00035     PUSHL   MEN$GA_TEST_QUEUE+8         ;
         00000000G  EF    02  FB 0003B     CALLS   #2, CRD$DISPOSE         ;
         00000000'  EF  7C 00042     CLRQ    MEN$GA_TEST_QUEUE+4         ; 1261
         00000000'  EF  D4 00048     CLRL    MEN$GA_TEST_QUEUE           ; 1262
            52 00000000'  EF  D0 0004E 2$:   MOVL    MEN$GA_DDB_HEAD, DDB_PTR       ; 1274
            36  13 00055     BEQL    5$              ; 1276
            53    62  D0 00057 3$:   MOVL    (DDB_PTR), NEXT_DDB           ; 1280
        10  A2  D5 0005A     TSTL    16(DDB_PTR)             ; 1282
            0D  13 0005D     BEQL    4$              ;
        10  A2  DD 0005F     PUSHL   16(DDB_PTR)             ; 1283
        28  A2  DD 00062     PUSHL   40(DDB_PTR)             ;
         00000000G  EF    02  FB 00065     CALLS   #2, CRD$DISPOSE         ;
            52  DD 0006C 4$:   PUSHL   DDB_PTR              ; 1286
        24  A2  DD 0006E     PUSHL   36(DDB_PTR)             ;
         00000000G  EF    02  FB 00071     CALLS   #2, CRD$DISPOSE         ;
            52    53  D0 00078     MOVL    NEXT_DDB, DDB_PTR         ; 1289
            50 00000000'  EF  9E 0007B     MOVAB   MEN$GA_DDB_HEAD, R0          ; 1291
            50    52  D1 00082     CMPL    DDB_PTR, R0             ;
```

```
            D0  12 00085    BNEQ    3$              ;
        00000000'  EF  D4 00087    CLRL    MEN$GA_DDB_HEAD          ; 1295
        00000000'  EF  94 0008D 5$:   CLRB    MEN$GB_FNCTST_INIT           ; 1302
            50      01  D0 00093    MOVL    #1, R0              ; 1303
           04 00096    RET                 ; 1304

; Routine Size:  151 bytes,    Routine Base:  CODE + 076C


;   1305  1 END
;   1306  0 ELUDOM




;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          727  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  WORK           56  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE         2051  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



Symbol                            Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
$ASCIC    Macro    Lib02   188     189     237     330     379     435     477     542     850    1076
             1079    1082
$BITPOSITION   Macro    Lib04   896
$CLOSE    KeyWMacr Lib04   906
$CRD_LITERALS   Macro    Lib03   102
$CRD_MESSAGE   Macro    Lib03   839    1125    1183
$CRD_PRINT   Unbound          839    1125    1192
    Macro    Lib03   237     330     379     435     477     542     839     850    1125    1192
$DS_DSADEF   Macro    Lib02   104
$DS_HPO_DECL   Macro    Lib02   474     746    1069
$DS_HPO_F   Fieldset Lib02   474     746    1069
$DS_LOAD   KeyWMacr Lib02   919
$DS_TYPEDEF   Macro    Lib02   237     330     379     435     477     542     839     851    1125    1192
$EQULST    Macro    Lib04   102     103     237     330     379     435     477     542     839     851
             1125    1192
$ER    Comptime    99
$FAB    KeyWMacr Lib04   896
$FAB_DECL   Macro    Lib04   896
$FAB_INIT   KeyWMacr Lib04   900
$MEN_DBASEID_ATTR  Macro    Lib03   471
$MEN_DBASEID_FLD  Fieldset Lib03   471
$MEN_DIAGBLK_ATTR  Macro    Lib03   590
$MEN_DIAGBLK_FLD  Fieldset Lib03   590
$MEN_FUNCSUP_DATBASE_ATTR Macro    Lib03   134     750
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------- -------- ----- ---------------
$MEN_FUNCSUP_DATBASE_FLD Fieldset Lib03   134     750
$MEN_LITERALS   Macro    Lib03   103
$MEN_QUEUE_LINK_ATTR  Macro    Lib03   128    1234    1235
$MEN_QUEUE_LINK_FLD  Fieldset Lib03   128    1234    1235
$MEN_SUPBLK_ATTR  Macro    Lib03   468     593     741     966    1067
$MEN_SUPBLK_FLD   Fieldset Lib03   468     593     741     966    1067
$MEN_TEST_QUEUE_ATTR  Macro    Lib03   152
$MEN_TSTCNFG_ATTR  Macro    Lib03   592
$MEN_TSTCNFG_FLD  Fieldset Lib03   592
$MODULE    Bind         99
$OPEN    KeyWMacr Lib04   904
$RMS_BITFLD   Macro    Lib04   896
$RMS_BITFLD_INI   Macro    Lib04   902
$RMS_BITS   Unbound           896
    Macro    Lib04   896     902
$RMS_CODFLD   Macro    Lib04   896
$RMS_CODFLD_INI   Macro    Lib04   902
$RMS_OR    Macro    Lib04   896     902
$RMS_PTR   Bind        902   902=
$RMS_VALFLD   Unbound           896
    Macro    Lib04   896     897
$RMS_VALFLD_INI   Macro    Lib04   902
$XABFHC    KeyWMacr Lib04   897
$XABFHC_DECL   Macro    Lib04   897
ADDR    MacrForm   108   110
ADDRESS    Bind        833   845=
AUTO$K_EXIT_AUTOSIZER_ERROR Literal    102
AUTO$K_EXIT_CONTROL_C  Literal    102
AUTO$K_EXIT_DUMMY_2  Literal    102
AUTO$K_EXIT_DUMMY_3  Literal    102
AUTO$K_EXIT_ERRORS_COMPLETE Literal    102
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal    102
AUTO$K_EXIT_INTERNAL_ERROR Literal    102
AUTO$K_EXIT_INV_BASE_SYSTEM Literal    102
AUTO$K_EXIT_LAST_REASON  Literal    102   102
AUTO$K_EXIT_NOERRORS_COMPLETE Literal    102
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal    102
AUTO$K_EXIT_NOERRORS_PREP Literal    102
A_FILE_DEFAULT   Own        185   486=    487=    495a
A_MENFUNCSUP_FILE  Own        183   483=    484=    494a
CRD$ALLOCATE_MEMORY  BindRout   828   836c
CRD$COMMON_INITIALIZATIONS ExtRout  Lib03   252c
CRD$DISPOSE   ExtRout  Lib03  1243c   1258c   1283c   1286c
CRD$EXE$ALONONPAGED  External Lib03   828.
CRD$GA_PTABLE   External Lib03   375.    376.
CRD$GB_CRD_DEBUG  External Lib03   236.    329.    378.    434.    476.    541.    849.
CRD$GB_CURRENT_STATE_TABLE External Lib03   307=
CRD$GT_NEW_LINE_MESSAGE  External Lib03   839a   1125a   1192a
CRD$GT_STAR_LINE_PREFIX_MESSAGE Unbound          839    1125    1192
CRD$GT_STAR_LINE_SUFFIX_MESSAGE Unbound          839    1125    1192
CRD$INIT_GENERAL_COM_TABLE ExtRout  Lib03   309c
CRD$K_ACTION_EQL_DO_NOTHING Literal    102
CRD$K_ACTION_RANGE_ERROR Literal    102
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_ADVANCE_DIAGNOSTIC Literal    102
CRD$K_ADVANCE_GENERAL_COM Literal    102
CRD$K_ADVANCE_STATE  Literal    102
CRD$K_ALLOCATION_ERROR  Literal    102   405     426     917
CRD$K_ASSIGN_PTABLE_PTRS Literal    102
CRD$K_AUTOSIZER_ERROR  Literal    102
CRD$K_AUTOSIZER_SET_FLAGS Literal    102
CRD$K_BAD_ACTION_NUMBER  Literal    102
CRD$K_BAD_TYPECODE_ERROR Literal    102
CRD$K_BASE_SYSTEM_IDC  Literal    102
CRD$K_BASE_SYSTEM_LESI  Literal    102
CRD$K_BASE_SYSTEM_NULL  Literal    102
CRD$K_BASE_SYSTEM_UDA_0  Literal    102
CRD$K_BASE_SYSTEM_UDA_1  Literal    102
CRD$K_BASE_SYSTEM_UDA_2  Literal    102
CRD$K_BASE_SYSTEM_UDA_3  Literal    102
CRD$K_CRD_INITIALIZATION Literal    102
CRD$K_CREATE_HST  Literal    102   537
CRD$K_DATA_LENGTH_ERROR  Literal    102
CRD$K_DDB_AUTO_SUPPORT_ENTRY Literal    102
CRD$K_DDB_BLINK   Literal    102
CRD$K_DDB_FLINK   Literal    102   515     759    1280
CRD$K_DDB_LAST_ENTRY  Literal    102   403     405
CRD$K_DDB_MEMORY_SIZE  Literal    102   409    1286
CRD$K_DDB_MENU_DEVICE_NUMB Literal    102   769
CRD$K_DDB_MENU_SUPPORT_ENTRY Literal    102   618    1107    1282    1283
CRD$K_DDB_MENU_SUPPORT_SIZE Literal    102   621    1283
CRD$K_DDB_NUMB_SUPPORT_TABLES Literal    102   624    1110
CRD$K_DDB_PTABLE_ENTRY  Literal    102   408     517     761
CRD$K_DDB_STATUS  Literal    102
CRD$K_DDB_SUPBLK_BLOCK_PTR Literal    102   529     600     762
CRD$K_DEVICE_NAME  Literal    102  1149    1153
CRD$K_DIAGNOSTIC_ERROR  Literal    102
CRD$K_DIAGNOSTIC_NAME  Literal    102   663    1131    1135
CRD$K_DONE_GENERAL_COMS  Literal    102
CRD$K_DO_NOTHING  Literal    102
CRD$K_DUMMY_10   Literal    102
CRD$K_DUMMY_11   Literal    102
CRD$K_DUMMY_12   Literal    102
CRD$K_DUMMY_13   Literal    102
CRD$K_DUMMY_3   Literal    102
CRD$K_DUMMY_4   Literal    102
CRD$K_DUMMY_5   Literal    102
CRD$K_DUMMY_6   Literal    102
CRD$K_DUMMY_7   Literal    102
CRD$K_DUMMY_8   Literal    102
CRD$K_DUMMY_9   Literal    102
CRD$K_EXIT_CRD   Literal    102
CRD$K_FAILURE   Literal  Lib03   616     840    1022
CRD$K_HOOK_POINT  Literal    102
CRD$K_HS_INTERNAL_ERROR  Literal    102
CRD$K_IDC_EVDLB   Literal    102
CRD$K_IDC_EVDLC   Literal    102
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_IDC_EVDLD   Literal    102
CRD$K_IDC_EVRAD_0  Literal    102
CRD$K_IDC_EVRAD_1  Literal    102
CRD$K_IDC_LAST_DIAGNOSTIC Literal    102
CRD$K_INTERNAL_ERROR  Literal    102
CRD$K_LAST_ACTION_ROUTINE Literal    102
CRD$K_LAST_ENTRY  Literal    102   614     709    1072    1194
CRD$K_LAST_FUNCTION  Literal    102
CRD$K_LAST_HOOK_POINT  Literal    102
CRD$K_LAST_INTERNAL_ERROR Literal    102
CRD$K_LAST_TYPE   Literal    102
CRD$K_LESI_ENWCA  Literal    102
CRD$K_LESI_EVRMB_0  Literal    102
CRD$K_LESI_EVRMB_1  Literal    102
CRD$K_LESI_LAST_DIAGNOSTIC Literal    102
CRD$K_LEVEL_4_PASSED  Literal    102
CRD$K_LOAD_AUTOSIZER  Literal    102
CRD$K_LOAD_ERROR  Literal    102
CRD$K_LOAD_GENERAL_COM  Literal    102
CRD$K_LOAD_LOAD_COM  Literal    102
CRD$K_LOAD_SELECT_COM  Literal    102
CRD$K_LOAD_SET_EVENT_COM Literal    102
CRD$K_LOAD_SET_FLAGS_COM Literal    102
CRD$K_LOAD_START_COM  Literal    102
CRD$K_LOAD_SUP_FILE  Literal    102
CRD$K_MM_INTERNAL_ERROR  Literal    102
CRD$K_NEW_LINE   Literal    102
CRD$K_PREPARATION_ERROR  Literal    102
CRD$K_PREPARATION_ERROR_TEXT Literal    102  1173    1177
CRD$K_RUNTIME   Literal    102   703    1167    1171
CRD$K_SAME_LINE   Literal    102
CRD$K_SET_EVENT_ARGS  Literal    102   679    1143    1147
CRD$K_SET_FLAGS_ARGS  Literal    102   671    1137    1141
CRD$K_SET_UP_DIAGNOSTIC  Literal    102
CRD$K_START   Literal    102
CRD$K_START_AUTOSIZER  Literal    102
CRD$K_START_ERROR  Literal    102
CRD$K_START_QUALIFIERS  Literal    102   687    1155    1159
CRD$K_STAR_LINE   Literal    102
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal    102
CRD$K_STATE_ADVANCE_GENERAL_COM Literal    102
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal    102
CRD$K_STATE_AUTOSIZER_ERROR Literal    102
CRD$K_STATE_AUTOSIZER_RUNNING Literal    102
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal    102
CRD$K_STATE_DIAGNOSTIC_ERROR Literal    102
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal    102
CRD$K_STATE_DONE_DIAGNOSTICS Literal    102
CRD$K_STATE_DONE_GENERAL_COMS Literal    102
CRD$K_STATE_DUMMY_1  Literal    102
CRD$K_STATE_DUMMY_10  Literal    102
CRD$K_STATE_DUMMY_12  Literal    102
CRD$K_STATE_DUMMY_13  Literal    102
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ ------ -------- ----- ---------------
CRD$K_STATE_DUMMY_2  Literal     102
CRD$K_STATE_DUMMY_3  Literal     102
CRD$K_STATE_DUMMY_4  Literal     102
CRD$K_STATE_DUMMY_6  Literal     102
CRD$K_STATE_DUMMY_7  Literal     102
CRD$K_STATE_DUMMY_8  Literal     102
CRD$K_STATE_EXIT_CRD  Literal     102
CRD$K_STATE_INTERNAL_ERROR Literal     102
CRD$K_STATE_LAST_STATE  Literal     102
CRD$K_STATE_LEVEL_4_PASSED Literal     102
CRD$K_STATE_LOAD_AUTOSIZER Literal     102
CRD$K_STATE_LOAD_ERROR  Literal     102
CRD$K_STATE_LOAD_GENERAL_COM Literal     102
CRD$K_STATE_LOAD_LOAD_COM Literal     102
CRD$K_STATE_LOAD_SELECT_COM Literal     102
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     102
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     102
CRD$K_STATE_LOAD_START_COM Literal     102
CRD$K_STATE_PREPARATION_ERROR Literal     102
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     102
CRD$K_STATE_START  Literal     102
CRD$K_STATE_START_AUTOSIZER Literal     102
CRD$K_STATE_START_ERROR  Literal     102
CRD$K_SUCCESS   Literal  Lib03   332     437     544     714     787     853     931    1026    1202    1303
CRD$K_TEST_START_TEXT  Literal     102   695    1161    1165
CRD$K_TQ_INTERNAL_ERROR  Literal     102
CRD$K_TYPECODE   Literal     102
CRD$K_UDA_0_EVDLB  Literal     102
CRD$K_UDA_0_EVDLC  Literal     102
CRD$K_UDA_0_EVDLD  Literal     102
CRD$K_UDA_0_EVRLA_0  Literal     102
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     102
CRD$K_UDA_1_EVDLB  Literal     102
CRD$K_UDA_1_EVDLC  Literal     102
CRD$K_UDA_1_EVDLD  Literal     102
CRD$K_UDA_1_EVRLA_0  Literal     102
CRD$K_UDA_1_EVRLA_1  Literal     102
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     102
CRD$K_UDA_2_EVDLB  Literal     102
CRD$K_UDA_2_EVDLC  Literal     102
CRD$K_UDA_2_EVDLD  Literal     102
CRD$K_UDA_2_EVRLA_0  Literal     102
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     102
CRD$K_UDA_3_EVDLB  Literal     102
CRD$K_UDA_3_EVDLC  Literal     102
CRD$K_UDA_3_EVDLD  Literal     102
CRD$K_UDA_3_EVRLA_0  Literal     102
CRD$K_UDA_3_EVRLA_1  Literal     102
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     102
CRD$PRINT   External Lib03   237ac   330ac   379ac   435ac   477ac   542ac   839ac   851ac  1125ac  1192ac
CRD$STOP   Unbound          111     119     121
    ExtRout  Lib03   260c    325c    405c    426c    537c    904c    917c    929c
CREATE_HST   Routine    548    80f    535c
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------------- -------- ----- ---------------
DATABASE   Bind        971   976.    977.
DATBASE_ID   Register   471   500=    505.    505a.
DBASEID$L_SIZE   Field    Lib03   505.
DDB    Bind        597   599a
DDB_COUNT   Register   370   393=    410=    410.    418.
DDB_PTR    Local       364   403a    405.    407.    408a=   409a=
    Register   473   513=    515=    515a.   517a.   529a=   535.    537.
    Register   594   599=    600a.   618a=   621a=   624a=
    Local      745   758=    759=    759a.   761a.   762a.   769a=
    Local     1068  1105=   1107a.  1110a.  1119.
    Local     1233  1274=   1276.   1280a.  1282a.  1283a.  1286a.  1286.   1289=   1291.
DDB_SIZE   Local       366   403a    405.    409.
DDB_TSTQ_TBL_ENTRIES  Register   369   418=    421.    426.
DEVICE_DATA_BLOCK_STRUCTURE Structur Lib03   364     473     594     745     1068    1233
DEVICE_NAME_PTR   Register   472   518=    525.
    Local     1059  1151=   1153=   1192.
DEVNAM    Bind        972   983.    984.
DIAGBLK$B_FRST_RECORD  Field    Lib03   657a
DIAGBLK$W_RECORD_NO  Field    Lib03   655.
DIAGBLK_INDEX   Register   589   636=    652=    652.
DIAGBLK_NO   Register   588   606=    614.    633=    652.
DIAGBLK_PTR   Register   590   638=    655a.   657aa   707=
DIAGNOSTIC_NAME_PTR  Local      1056  1133=   1135=   1192.
DS$K_PRINTB   Literal    237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_PRINTF   Literal     237   237
    Literal     330   330
    Literal     379   379
    Literal     435   435
    Literal     477   477
    Literal     542   542
    Literal     839   839
    Literal     839   839
    Literal     851   851
    Literal    1125  1125
    Literal    1125  1125
    Literal    1192  1192
    Literal    1192  1192
DS$K_PRINTI   Literal     237
    Literal     330
    Literal     379
    Literal     435
```

Symbol	Type	Defined	Referenced ...
	Literal	477	
	Literal	542	
	Literal	839	
	Literal	839	
	Literal	851	
	Literal	1125	
	Literal	1125	
	Literal	1192	
	Literal	1192	
DS$K_PRINTX	Literal	237	
	Literal	330	
	Literal	379	
	Literal	435	
	Literal	477	
	Literal	542	
	Literal	839	
	Literal	839	
	Literal	851	
	Literal	1125	
	Literal	1125	
	Literal	1192	
	Literal	1192	
DS$K_TYPE_ABORT_PROGRAM	Literal	237	
	Literal	330	
	Literal	379	
	Literal	435	
	Literal	477	
	Literal	542	
	Literal	839	
	Literal	839	
	Literal	851	
	Literal	1125	
	Literal	1125	
	Literal	1192	
	Literal	1192	
DS$K_TYPE_ABORT_TEST	Literal	237	
	Literal	330	
	Literal	379	
	Literal	435	
	Literal	477	
	Literal	542	
	Literal	839	
	Literal	839	
	Literal	851	
	Literal	1125	
	Literal	1125	
	Literal	1192	
	Literal	1192	
DS$K_TYPE_COMMAND_ERR	Literal	237	
	Literal	330	
	Literal	379	
	Literal	435	
	Literal	477	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_COMMAND_OUT  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_CRD_AUTOTEST  Literal     237   237
    Literal     330   330
    Literal     379   379
    Literal     435   435
    Literal     477   477
    Literal     542   542
    Literal     839   839
    Literal     839   839
    Literal     851   851
    Literal    1125  1125
    Literal    1125  1125
    Literal    1192  1192
    Literal    1192  1192
DS$K_TYPE_DS_PROMPT  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_DS_START  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------

    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_ERRDEV  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_ERRHARD  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_ERROR_BODY  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_ERROR_END  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_ERRPREP  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_ERRSOFT  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_ERRSUP  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_ERRSYS  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_ERR_HALT  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_EXCEPTION  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_EXCEPTION_HEAD Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_FIRST_PASS  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_GENERAL  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_GENERAL_ERROR  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_NO_TESTS  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_PARAM_ERROR  Literal     237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal   1125
    Literal   1192
    Literal   1192
DS$K_TYPE_PROGRAM_END  Literal    237
    Literal    330
    Literal    379
    Literal    435
    Literal    477
    Literal    542
    Literal    839
    Literal    839
    Literal    851
    Literal   1125
    Literal   1125
    Literal   1192
    Literal   1192
DS$K_TYPE_PROGRAM_INFO  Literal    237
    Literal    330
    Literal    379
    Literal    435
    Literal    477
    Literal    542
    Literal    839
    Literal    839
    Literal    851
    Literal   1125
    Literal   1125
    Literal   1192
    Literal   1192
DS$K_TYPE_PROGRAM_START  Literal    237
    Literal    330
    Literal    379
    Literal    435
    Literal    477
    Literal    542
    Literal    839
    Literal    839
    Literal    851
    Literal   1125
    Literal   1125
    Literal   1192
    Literal   1192
DS$K_TYPE_QIO_INVADP  Literal    237
    Literal    330
    Literal    379
    Literal    435
    Literal    477
    Literal    542
    Literal    839
    Literal    839
    Literal    851
    Literal   1125
    Literal   1125
```

Symbol	Type	Defined	Referenced ...
	Literal	1192	
	Literal	1192	
DS$K_TYPE_QIO_NODRIVER	Literal	237	
	Literal	330	
	Literal	379	
	Literal	435	
	Literal	477	
	Literal	542	
	Literal	839	
	Literal	839	
	Literal	851	
	Literal	1125	
	Literal	1125	
	Literal	1192	
	Literal	1192	
DS$K_TYPE_QIO_WRONGVER	Literal	237	
	Literal	330	
	Literal	379	
	Literal	435	
	Literal	477	
	Literal	542	
	Literal	839	
	Literal	839	
	Literal	851	
	Literal	1125	
	Literal	1125	
	Literal	1192	
	Literal	1192	
DS$K_TYPE_SCRIPT_ECHO	Literal	237	
	Literal	330	
	Literal	379	
	Literal	435	
	Literal	477	
	Literal	542	
	Literal	839	
	Literal	839	
	Literal	851	
	Literal	1125	
	Literal	1125	
	Literal	1192	
	Literal	1192	
DS$K_TYPE_SCRIPT_PNF	Literal	237	
	Literal	330	
	Literal	379	
	Literal	435	
	Literal	477	
	Literal	542	
	Literal	839	
	Literal	839	
	Literal	851	
	Literal	1125	
	Literal	1125	
	Literal	1192	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal   1192
DS$K_TYPE_SCRIPT_PROMPT  Literal    237
    Literal    330
    Literal    379
    Literal    435
    Literal    477
    Literal    542
    Literal    839
    Literal    839
    Literal    851
    Literal   1125
    Literal   1125
    Literal   1192
    Literal   1192
DS$K_TYPE_SCRIPT_SKIP  Literal    237
    Literal    330
    Literal    379
    Literal    435
    Literal    477
    Literal    542
    Literal    839
    Literal    839
    Literal    851
    Literal   1125
    Literal   1125
    Literal   1192
    Literal   1192
DS$K_TYPE_SEQUENCE_ERROR Literal    237
    Literal    330
    Literal    379
    Literal    435
    Literal    477
    Literal    542
    Literal    839
    Literal    839
    Literal    851
    Literal   1125
    Literal   1125
    Literal   1192
    Literal   1192
DS$K_TYPE_START_ERR  Literal    237
    Literal    330
    Literal    379
    Literal    435
    Literal    477
    Literal    542
    Literal    839
    Literal    839
    Literal    851
    Literal   1125
    Literal   1125
    Literal   1192
    Literal   1192

.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DS$K_TYPE_START_LIST  Literal    237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_SUMMARY  Literal    237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$K_TYPE_USER_PROMPT  Literal    237
    Literal     330
    Literal     379
    Literal     435
    Literal     477
    Literal     542
    Literal     839
    Literal     839
    Literal     851
    Literal    1125
    Literal    1125
    Literal    1192
    Literal    1192
DS$LOAD    ExtRout    926   926c
DSA$AL_APTMAIL   Bind        104   104
DSA$GL_APTCOM    Bind        104
DSA$GL_DEVLEN    Bind        104
DSA$GL_DEVNAM    Bind        104
DSA$GL_ERRNO    Bind        104
DSA$GL_EVENT    Bind        104
DSA$GL_FLAGS    Bind        104   254     258     259     320     323     324     998    1004    1010
DSA$GL_MSGPTR    Bind        104
DSA$GL_MSGTYP    Bind        104
DSA$GL_PASSES    Bind        104
DSA$GL_PASSNO    Bind        104
DSA$GL_SECTNO    Bind        104
DSA$GL_SID   Bind        104
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
DSA$GL_SUBTNO    Bind        104
DSA$GL_TESTNO    Bind        104
DSA$GL_UNITS   Bind        104
DSA$V_BINARY    Macro    Lib02   254     320
DSA$V_CRD_MENUTEST_OFF  Macro    Lib02   258     323     1010
DSA$V_CRD_MENUTEST_ON  Macro    Lib02   259     324     998    1004
DSA_EXTRACT    RoutForm   192   256.    321.
DSC$A_POINTER    Macro    Lib04   484     487
DSC$K_S_BLN   Literal  Lib04   183     185
DSC$W_LENGTH    Macro    Lib04   483     486
END_OF_DATABASE   Local       968   976=    978.
FAB    Local       896   896=    902a    904a    906a    910.    911.    912.
FAB$B_BID    Macro    Lib04   902
FAB$B_BKS    Macro    Lib04   902
FAB$B_BLN    Macro    Lib04   902
FAB$B_DNS    Macro    Lib04   902
FAB$B_FAC    Macro    Lib04   902
FAB$B_FNS    Macro    Lib04   902
FAB$B_FSZ    Macro    Lib04   902     912
FAB$B_ORG    Macro    Lib04   902
FAB$B_RAT    Macro    Lib04   902
FAB$B_RFM    Macro    Lib04   902     911
FAB$B_RTV    Macro    Lib04   902
FAB$B_SHR    Macro    Lib04   902
FAB$C_BID    Literal  Lib04   896     902
FAB$C_BLN    Literal  Lib04   896     902
FAB$C_SEQ    Literal  Lib04   896
FAB$C_VAR    Literal  Lib04   896     902
FAB$L_ALQ    Macro    Lib04   902
FAB$L_CTX    Macro    Lib04   902
FAB$L_DNA    Macro    Lib04   902
FAB$L_FNA    Macro    Lib04   902
FAB$L_FOP    Macro    Lib04   902
FAB$L_JNL    Macro    Lib04   902
FAB$L_MRN    Macro    Lib04   902
FAB$L_NAM    Macro    Lib04   902
FAB$L_XAB    Macro    Lib04   902
FAB$M_BIO    Literal  Lib04   902
FAB$M_GET    Literal  Lib04   896     902
FAB$V_CHAN_MODE    Macro    Lib04   896     902
FAB$V_FILE_MODE    Macro    Lib04   896     902
FAB$V_LNM_MODE    Macro    Lib04   896     902
FAB$W_BLS    Macro    Lib04   902
FAB$W_DEQ    Macro    Lib04   902
FAB$W_GBC    Macro    Lib04   902
FAB$W_MRS    Macro    Lib04   902     910
FALSE    Literal  Lib03   168     175     244     975    1302
FILE_ATTRIBUTES   Local       895   910=    911=    912=    926a
GET1ST_    Macro    Lib04   102     103     237     330     379     435     477     542     839     851
             1125    1192
GET2ND_    Unbound          102     103     237     330     379     435     477     542     839     851
             1125    1192
HP$K_LENGTH   Literal  Lib02   474     746    1069
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HP$T_TYPE   Field    Lib02   518a
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal    102
HS$K_ACTION_ADVANCE_GENERAL_COM Literal    102
HS$K_ACTION_ADVANCE_STATE Literal    102
HS$K_ACTION_CHANGE_TABLE Literal    102
HS$K_ACTION_DIAGNOSTIC_ERROR Literal    102
HS$K_ACTION_DONE_GENERAL_COMS Literal    102
HS$K_ACTION_DO_NOTHING  Literal    102
HS$K_ACTION_DUMMY_3  Literal    102
HS$K_ACTION_DUMMY_4  Literal    102
HS$K_ACTION_DUMMY_5  Literal    102
HS$K_ACTION_DUMMY_6  Literal    102
HS$K_ACTION_INIT_HS  Literal    102
HS$K_ACTION_INTERNAL_ERROR Literal    102
HS$K_ACTION_LAST_ACTION  Literal    102
HS$K_ACTION_LOAD_ERROR  Literal    102
HS$K_ACTION_LOAD_GENERAL_COM Literal    102
HS$K_ACTION_LOAD_LOAD_COM Literal    102
HS$K_ACTION_LOAD_SELECT_COM Literal    102
HS$K_ACTION_LOAD_SET_EVENT_COM Literal    102
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal    102
HS$K_ACTION_LOAD_START_COM Literal    102
HS$K_ACTION_PREPARATION_ERROR Literal    102
HS$K_ACTION_START_ERROR  Literal    102
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal    102
HS$K_STATE_ADVANCE_GENERAL_COM Literal    102
HS$K_STATE_CHANGE_TABLE  Literal    102
HS$K_STATE_DIAGNOSTIC_ERROR Literal    102
HS$K_STATE_DIAGNOSTIC_RUNNING Literal    102
HS$K_STATE_DONE_GENERAL_COMS Literal    102
HS$K_STATE_DUMMY_1  Literal    102
HS$K_STATE_DUMMY_2  Literal    102
HS$K_STATE_DUMMY_3  Literal    102
HS$K_STATE_DUMMY_4  Literal    102
HS$K_STATE_DUMMY_6  Literal    102
HS$K_STATE_INIT_HS  Literal    102
HS$K_STATE_INTERNAL_ERROR Literal    102
HS$K_STATE_LAST_STATE  Literal    102
HS$K_STATE_LOAD_ERROR  Literal    102
HS$K_STATE_LOAD_GENERAL_COM Literal    102
HS$K_STATE_LOAD_LOAD_COM Literal    102
HS$K_STATE_LOAD_SELECT_COM Literal    102
HS$K_STATE_LOAD_SET_EVENT_COM Literal    102
HS$K_STATE_LOAD_SET_FLAGS_COM Literal    102
HS$K_STATE_LOAD_START_COM Literal    102
HS$K_STATE_PREPARATION_ERROR Literal    102
HS$K_STATE_START_ERROR  Literal    102
HST_INDEX   Local     1071  1111=   1113=   1113.
HST_NO    Local     1070  1110=   1113.
HST_PTR    Local      581   614a    618.    663a=   671a=   679a=   687a=   695a=   703a=   709=    709aa
    Local     1072  1107=   1119.   1131a.  1135a.  1137a.  1141a.  1143a.  1147a.  1149a.  1153a.
           1155a.  1159a.  1161a.  1165a.  1167a.  1171a.  1173a.  1177a.  1194=   1194.
HST_SIZE   Local      580   614a    621.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
INSQUE    Builtin     71   407
INTERNAL_ERROR_EXIT  Macro      108   405     425     537     917
I_ADDRESS   RoutForm   791   833.
    RoutForm   857   915.    917a.   926a.
I_DDB    RoutForm   548   597.
I_DEVNAM   RoutForm   935   972.
I_MEMSIZE   RoutForm   857   915.    917a.
I_RETSIZE   RoutForm   791   834.
I_SIZE    RoutForm   857   908a=   915.    917a.   926a.   926.
I_SUPPORT_BLK   RoutForm   935   973.
JSB_ALLOC   Linkage  Lib01   828
LOAD_ERROR_EXIT   Macro      116   904     929
MEN$ALLOCATE_MEMORY  GlobRout   791 Lib03e     82f    403c    421c    614c    915c
MEN$BUILD_DDB   GlobRout   336 Lib03e     78f
MEN$CNTLCINIT_TBL_INIT  ExtRout  Lib03   292c
MEN$CNTLC_MENU   ExtRout  Lib03   252a
MEN$CNTLC_TBL_INIT  ExtRout  Lib03   298c
MEN$DETERMINE_SUPPORT  GlobRout   441 Lib03e     79f
MEN$DUMP_TSTQ_HST  GlobRout  1030 Lib03e     85f
MEN$FIND_SUPBLK   GlobRout   935 Lib03e     84f    524c
MEN$FNCTST_TESTSTRT_TEXT ExtRout  Lib03  1119c
MEN$FTMTBL_INIT   ExtRout  Lib03   280c
MEN$GA_DDB_HEAD   Global     128 Lib03e    130=    131=    385=    385a    386=    386a    407.    513a    515
            758a    759    1274.   1291    1294=   1295=
MEN$GA_FUNCSUP_DATBASE  Global     134 Lib03e    136=    137=    138=    139=    496a    497a    498a    500.    504=
            524a    750a   1241.   1243.   1244.   1246=   1247=   1248=   1249=
MEN$GA_TEST_QUEUE  Global     152 Lib03e    154=    155=    156=    422a    426.    428=    431=   1097.   1098.
            1256.   1258.   1260=   1261=   1262=
MEN$GB_FNCTST_INIT  Global     168 Lib03e    168=    244=   1302=
MEN$GB_TEST_IN_PROGRESS  Global     175 Lib03e    175=
MEN$GK_FUNCSUP_DATBASE  Literal  Lib03   134     750
MEN$GK_TEST_QUEUE  Literal    103
MEN$GT_FILE_DEFAULT  External Lib03
    GlobBind   189   486.    487a
MEN$GT_MENFUNCSUP_FILE  External Lib03
    GlobBind   188   483.    484a
MEN$GT_NOALL_NONPGMEM  External Lib03   839a
MEN$INIT_HS_STATE_TABLE  ExtRout  Lib03   312c
MEN$INIT_MM_STATE_TABLE  ExtRout  Lib03   310c
MEN$INIT_TQ_STATE_TABLE  ExtRout  Lib03   311c
MEN$K_CNTLC_SEL_CONTINUE Literal    103
MEN$K_CNTLC_SEL_EXIT  Literal    103
MEN$K_CNTLC_SEL_FTMENU  Literal    103
MEN$K_CNTLC_SEL_LAST_ENTRY Literal    103
MEN$K_DBASEID_SIZE  Literal  Lib03   471
MEN$K_DEVTSTPREP_1  Literal    103
MEN$K_DEVTSTPREP_2  Literal    103
MEN$K_DEVTSTPREP_3  Literal    103
MEN$K_DEVTSTPREP_4  Literal    103
MEN$K_DEVTSTPREP_5  Literal    103
MEN$K_DEVTSTPREP_6  Literal    103
MEN$K_DEVTSTPREP_7  Literal    103
MEN$K_DEVTSTPREP_NULL  Literal    103
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$K_DIAGBLK_SIZE  Literal  Lib03   590
MEN$K_FTMRET_EXIT  Literal    103
MEN$K_FTMRET_FAILURE  Literal    103
MEN$K_FTMRET_MENU  Literal    103
MEN$K_FTMRET_TEST  Literal    103
MEN$K_FTMSEL_EXIT  Literal    103
MEN$K_FTMSEL_FNCTST_ALL  Literal    103
MEN$K_FTMSEL_LAST_ENTRY  Literal    103
MEN$K_FTMSEL_PREP  Literal    103
MEN$K_FTMSEL_SYSTEM_HDWR Literal    103
MEN$K_FTMSEL_TEST  Literal    103
MEN$K_HS_STATE_TABLE  Literal    102
MEN$K_MM_STATE_TABLE  Literal    102   307
MEN$K_PREP_ERROR_TEXT_LEN Literal    103
MEN$K_QUEUE_LINK_SIZE  Literal  Lib03   128    1234    1235
MEN$K_SELDEV_NUMB_START  Literal    103   757
MEN$K_SUPBLK_SIZE  Literal  Lib03   468     593     741     966    1067
MEN$K_TEST_QUEUE_PTR_BLK_SIZE Literal  Lib03   152
MEN$K_TMENU_TSTALL_DEVNUMB Literal    103
MEN$K_TQ_STATE_TABLE  Literal    102
MEN$K_TSTCLA_ALL  Literal    103
MEN$K_TSTCLA_CARD  Literal    103   784
MEN$K_TSTCLA_COMM  Literal    103   780
MEN$K_TSTCLA_CPU  Literal    103   776
MEN$K_TSTCLA_DISK  Literal    103   778
MEN$K_TSTCLA_IO_ADAPTOR  Literal    103   785
MEN$K_TSTCLA_LAST_ENTRY  Literal    103
MEN$K_TSTCLA_MEMORY  Literal    103   777
MEN$K_TSTCLA_NULL  Literal    103
MEN$K_TSTCLA_PRINTER  Literal    103   783
MEN$K_TSTCLA_REAL  Literal    103   781
MEN$K_TSTCLA_TAPE  Literal    103   779
MEN$K_TSTCLA_TERM  Literal    103   782
MEN$K_TSTCNFG_SIZE  Literal  Lib03   592
MEN$K_TSTCTG_CPU  Literal    103
MEN$K_TSTCTG_IO_ADAPTOR  Literal    103
MEN$K_TSTCTG_IO_CONTROLLER Literal    103
MEN$K_TSTCTG_IO_UNIT  Literal    103
MEN$K_TSTCTG_MEMORY  Literal    103
MEN$K_TSTCTG_NULL  Literal    103
MEN$K_TSTTXT_DEVNAM_SZ  Literal    103
MEN$K_TSTTXT_TSTCLA_SZ  Literal    103
MEN$K_TSTTXT_UUTNAM_SZ  Literal    103
MEN$LOAD_FILE   GlobRout   857 Lib03e     83f     94c
MEN$MENU_CLEANUP  GlobRout  1206 Lib03e     86f
MEN$MENU_FNCTST_INIT  GlobRout   192 Lib03e     77f
MEN$MENU_TEST_INTERNAL_ERROR Unbound          110
    ExtRout  Lib03   405c    426c    537c    917c
MEN$PREPTBL_INIT  ExtRout  Lib03   286c
MEN$SPECIFY_SELECTNO  GlobRout   718 Lib03e     81f
MEN$V_FUNCSUP_DATBASE_PTR Field    Lib03   137=    504=   1241.   1247=
MEN$V_FUNCSUP_DATBASE_SIZE Field    Lib03   136=    496a   1248=
MEN$V_FUNCSUP_DATBASE_START Field    Lib03   139=    497a    500.   1244.   1249=
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$V_FUNCSUP_DATBASE_TOTAL_SIZ Field    Lib03   138=    498a   1243.   1246=
MEN$V_TEST_QUEUE_PTR_BLK_FLDS Fieldset Lib03   152
MEN$V_TQ_TABLE_NUMB_ENTRIES Field    Lib03   154=    428=   1097.   1262=
MEN$V_TQ_TABLE_PTR  Field    Lib03   155=    422a    426.   1098.   1256.   1258.   1261=
MEN$V_TQ_TABLE_SIZE  Field    Lib03   156=    431=   1258.   1260=
MENU$K_EXIT_AUTOSIZER_ERROR Literal    102
MENU$K_EXIT_INTERNAL_ERROR Unbound           111
    Literal    102   260     325     405     426     537     917
MENU$K_EXIT_LAST_REASON  Literal    102
MENU$K_EXIT_OPERATOR_ABORT Literal    102
MENU$K_EXIT_OPERATOR_STOP Literal    102
MENU$K_EXIT_SUP_FILE_LOAD_ERR Unbound           121
    Literal    102   904     929
MENU$K_EXIT_SUP_FILE_NOT_FOUND Unbound           119
    Literal    102   904     929
MM$K_ACTION_ADVANCE_STATE Literal    102
MM$K_ACTION_AUTOSIZER_ERROR Literal    102
MM$K_ACTION_BUILD_DDBS  Literal    102
MM$K_ACTION_BUILD_HSTS  Literal    102
MM$K_ACTION_CHANGE_TABLE Literal    102
MM$K_ACTION_DONE_INITS  Literal    102
MM$K_ACTION_DO_NOTHING  Literal    102
MM$K_ACTION_DUMMY_3  Literal    102
MM$K_ACTION_DUMMY_4  Literal    102
MM$K_ACTION_DUMMY_5  Literal    102
MM$K_ACTION_DUMMY_6  Literal    102
MM$K_ACTION_DUMMY_7  Literal    102
MM$K_ACTION_DUMMY_8  Literal    102
MM$K_ACTION_INTERNAL_ERROR Literal    102
MM$K_ACTION_LAST_ACTION  Literal    102
MM$K_ACTION_LOAD_AUTOSIZER Literal    102
MM$K_ACTION_MENU_PROMPT  Literal    102
MM$K_ACTION_PRINT_MENU  Literal    102
MM$K_ACTION_PRINT_MENU_HEADING Literal    102
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal    102
MM$K_ACTION_START_AUTOSIZER Literal    102
MM$K_STATE_AUTOSIZER_ERROR Literal    102
MM$K_STATE_AUTOSIZER_RUNNING Literal    102
MM$K_STATE_BUILD_DDBS  Literal    102
MM$K_STATE_BUILD_HSTS  Literal    102
MM$K_STATE_CHANGE_TABLE  Literal    102
MM$K_STATE_DONE_INITS  Literal    102
MM$K_STATE_DUMMY_1  Literal    102
MM$K_STATE_DUMMY_2  Literal    102
MM$K_STATE_DUMMY_3  Literal    102
MM$K_STATE_DUMMY_5  Literal    102
MM$K_STATE_DUMMY_7  Literal    102
MM$K_STATE_DUMMY_8  Literal    102
MM$K_STATE_INTERNAL_ERROR Literal    102
MM$K_STATE_LAST_STATE  Literal    102
MM$K_STATE_LOAD_AUTOSIZER Literal    102
MM$K_STATE_MENU_PROMPT  Literal    102
MM$K_STATE_PRINT_MENU  Literal    102
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_STATE_PRINT_MENU_HEADING Literal    102
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal    102
MM$K_STATE_START  Literal    102
MM$K_STATE_START_AUTOSIZER Literal    102
MODNAM    Macro    Lib01    99
NEXT_DDB   Local     1232  1280=   1289.
NUL2ND_    Macro    Lib04   102     103     237     330     379     435     477     542     839     851
           1125    1192
NUMB_PTABLE_ENTRIES  Bind       375   394.
ON    Literal  Lib03   254     258     259     320     323     324
PREPARATION_ERROR_TEXT_PTR Local     1063  1175=   1177=   1192.
PTABLE    Register   372   398=    400.    408.
PTABLE_PTR   Register   474   517=    518aa
    Local      746   761=
    Local     1069
PTABLE_TABLE   Bind       376   398.
PTABLE_TABLE_INDEX  Register   371   394=    396=    396.    398.
QUEUE$L_BLINK   Field    Lib03   131=    386=    407.
QUEUE$L_FLINK   Field    Lib03   130=    385=   1274.   1294=   1295=
QUEUE_BLKPTR   Local     1235
QUEUE_HEAD   Local     1234
RECORD_CNT   Register   585   654=    665=    665.    673=    673.    681=    681.    689=    689.    697=
            697.    705=    705.
RECORD_NO   Register   584   655=
RECORD_PTR   Register   586   657=    663.    664=    664a.   664aa   671.    672=    672a.   672aa   679.
            680=    680a.   680aa   687.    688=    688a.   688aa   695.    696=    696a.
            696aa   703.    704=    704a.   704aa   707.
REMQUE    Builtin     72
RETSIZE    Bind       834   846=
RETURNED_ADDRESS  Local      825   836=    844a=   845.    851.
RETURNED_SIZE   Local      824   836=    844.    846.    851.
RMS$_FNF   Unbound          118
    Literal  Lib04   904     929
RMS_STATUS   MacrForm   116   118
RUNTIME_PTR   Local     1062  1169=   1171=   1192.
R_DATABASE   RoutForm   935   971.
R_FILE    RoutForm   857   902.    902a.   926.
R_FILE_DEFAULT   RoutForm   857   902.    902a.   926.
R_SIZE    RoutForm   791   832.
R_TEST_CLASS   RoutForm   738   749.
SDL$STARLET_CONCAT  Macro    Lib04   904     906
SDL$STARLET_OPT   Macro    Lib04   904     906
SDL$STARLET_REQ   Macro    Lib04   904     906
SELECTNO_CLASS   Routine    738   776c    777c    778c    779c    780c    781c    782c    783c    784c    785c
SELECT_NO_INDEX   Local      744   757=    769.    770=    770.
SET_EVENT_ARGS_PTR  Local     1058  1145=   1147=   1192.
SET_FLAGS_ARGS_PTR  Local     1057  1139=   1141=   1192.
SIZE    Bind       832   836.
SIZE_1    MacrForm   108   110
SIZE_2    MacrForm   108   110
START_QUALIFIERS_PTR  Local     1060  1157=   1159=   1192.
STATUS    Local      894   904=    904.    919=    927.    929.
SUPBLK$B_STRTFRST_CNFGBLK Field    Lib03   603a
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
SUPBLK$B_TEST_CLASS  Field    Lib03   766.
SUPBLK$L_SUPBLK_SIZE  Field    Lib03  1016.
SUPBLK$T_DEVICE_NAME  Field    Lib03   985.    986
SUPBLK$V_STATUS_INVALID  Field    Lib03   991.
SUPBLK$W_LEVEL   Field    Lib03   996.   1002.   1008.
SUPBLK_FOUND   Local      967   975=    978.   1014=   1020.
SUPBLK_PTR   Local      468   526a    529.
    Register   593   600=    603aa
    Local      741   762=    763.    766a.
    Local      966   977=    978.    985a.   986.    991a.   996a.  1002a.  1008a.  1016=   1016a.
          1016.   1024.
    Local     1067
SUPPORT_BLK   Bind       973  1024=
SUPPORT_DATABASE  Bind        750
SYS$CLOSE   ExtRout    906   906c
SYS$OPEN   ExtRout    904   904c
TEST_CLASS   Bind       749   766.
TEST_START_TEXT_PTR  Local     1061  1163=   1165=   1192.
TQ$K_ACTION_ADVANCE_STATE Literal    102
TQ$K_ACTION_CHANGE_TABLE Literal    102
TQ$K_ACTION_DONE_TEST_QUEUE Literal    102
TQ$K_ACTION_DO_NOTHING  Literal    102
TQ$K_ACTION_DUMMY_3  Literal    102
TQ$K_ACTION_DUMMY_4  Literal    102
TQ$K_ACTION_DUMMY_5  Literal    102
TQ$K_ACTION_GET_DDB  Literal    102
TQ$K_ACTION_INIT_TQ  Literal    102
TQ$K_ACTION_INTERNAL_ERROR Literal    102
TQ$K_ACTION_LAST_ACTION  Literal    102
TQ$K_STATE_CHANGE_TABLE  Literal    102
TQ$K_STATE_DONE_TEST_QUEUE Literal    102
TQ$K_STATE_DUMMY_1  Literal    102
TQ$K_STATE_DUMMY_2  Literal    102
TQ$K_STATE_DUMMY_3  Literal    102
TQ$K_STATE_DUMMY_4  Literal    102
TQ$K_STATE_DUMMY_5  Literal    102
TQ$K_STATE_GET_DDB  Literal    102
TQ$K_STATE_INIT_TQ  Literal    102
TQ$K_STATE_INTERNAL_ERROR Literal    102
TQ$K_STATE_LAST_STATE  Literal    102
TRUE    Literal  Lib03  1014
TSTCNFG$B_DIAGBLK_NO  Field    Lib03   606.    624.    633.
TSTCNFG$B_STRTFRST_DIAGBLK Field    Lib03   638a
TSTCNFG_PTR   Register   592   603=    606a.   624a.   633a.   638aa
TSTQ_SIZE   Local      365   423a    426.    428.    431.
TSTQ_TBL_ENTRIES  Local     1065  1097=   1104.
TSTQ_TBL_INDEX   Local     1064  1096=   1104.   1105.   1199=   1199.
TSTQ_TBL_PTR   Local     1066  1098=   1105a.
TYPE    MacrForm   108   110
T_DUMP_HST   Bind      1081  1192a
T_HST_ENTRY_NULL  Bind      1075  1133a   1139a   1145a   1151a   1157a   1163a   1169a   1175a
T_HST_TITLE   Bind      1078  1125a
XAB    Local      897   897=    902a    908.
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
XAB$C_FHC   Literal  Lib04   897
XAB$C_FHCLEN   Literal  Lib04   897
XAB$L_EBK   Macro    Lib04   908
XAB$W_FFB   Macro    Lib04   908


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  --------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]MENTSTINI.B32;1
    2  Module MENTSTINI
   62  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   64  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   66  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   68  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
 1306  Eludom MENTSTINI


 KEY TO REFERENCE TYPE FLAGS

     . Fetch
     = Store
     c Routine call
     a Address use
     a Indirect use
     e External, external routine, or external literal declaration
     f Forward or forward routine declaration
     h Condition handler enabling
     m Map declaration
     u Undeclare declaration


;    Library Statistics
;
;           -------- Symbols --------     Pages Processing
; F  e          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671          2          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923          23         2          94          00:00.2
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486          95         19         62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619         64           0       1000           00:04.2
```

```
;     COMMAND QUALIFIERS

; BLISS MENTSTINI.B32/LIST=MENTSTINI.LIS/CROSS_REFERENCE/DEBUG/OBJECT=MENTSTINI.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  2051 code + 783 data bytes
; Run Time:    01:02.4
; Elapsed Time:    01:13.0
; Lines/CPU Min:    1255
; Lexemes/CPU-Min: 62185
; Memory Used: 384 pages
; Compilation Complete
```

```
;    0001  0 %TITLE  'CRD MENU - Test Queue Tables'
;    0002  0 MODULE MENTSTQ  (
;    0003  0   IDENT = '01-00'
;    0004  0   ) =
;    0005  0 !
;    0006  0 ! COPYRIGHT (C)  1982
;    0007  0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008  0 !
;    0009  0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010  0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011  0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012  0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013  0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014  0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015  0 ! REMAIN IN DEC.
;    0016  0 !
;    0017  0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018  0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019  0 ! CORPORATION.
;    0020  0 !
;    0021  0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022  0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023  0 !
```

```
;    0024  0 !++
;    0025  0 ! FACILITY:
;    0026  0 !
;    0027  0 ! ABSTRACT:
;    0028  0 !
;    0029  0 ! ENVIRONMENT:
;    0030  0 !
;    0031  0 ! AUTHOR:
;    0032  0 !
;    0033  0 ! John Ciukaj  May-1982
;    0034  0 !
;    0035  0 ! CREATION DATE:
;    0036  0 !
;    0037  0 ! VERSION:
;    0038  0 !
;    0039  0 ! Modified by:
;    0040  0 !
;    0041  0 !--
```

```
;    0042  0 %SBTTL 'Libraries'
;    0043  1 BEGIN
;    0044  1
;    0045  1
;    0046  1 LIBRARY '$DS';
;    0047  1 LIBRARY '$DIAG';
;    0048  1 LIBRARY '$CRD';
;    0049  1 LIBRARY 'SYS$LIBRARY:LIB';
```

```
;    0050  1 %SBTTL 'Forward-External  Routines'
;    0051  1
;    0052  1 FORWARD ROUTINE
;    0053  1     MEN$BldTstQ_FncTst_All : ADDRESSING_MODE (LONG_RELATIVE),
;    0054  1     MEN$BldTstQ_FncTst_Dev : ADDRESSING_MODE (LONG_RELATIVE);
```

```
;    0055  1 %SBTTL 'Psect Definitions'
;    0056  1
;    0057  1 PSECT
;    0058  1     Plit = Data (NoExecute, Share, NoWrite, Addressing_Mode (Long_Relative));
;    0059  1 PSECT
;    0060  1     Code =  Code (Execute, Share, NoWrite, Addressing_Mode (Long_Relative), PIC);
;    0061  1 PSECT
;    0062  1     Own  = Work (NoExecute, NoShare, Write, Addressing_Mode (Long_Relative));
;    0063  1 PSECT
;    0064  1     Global = Work (NoExecute, NoShare, Write, Addressing_Mode (Long_Relative));
```

```
;    0065  1 %SBTTL 'Module-Global Static Storage'
;    0066  1
;    0067  1      Modnam ('MENTSTQ');   ! Module Name for Errsup Macro
```

```
ZZ-ENSAB-2.0    Macro - Literal Definitions                    19-Sep-1985      Fiche 8  Frame F15        Sequence 1629
MENTSTQ  CRD MENU - Test Queue Tables   19-Sep-1985 17:24:51 VAX-11 Bliss-32 V4.1-746      Page   7
01-00  Macro - Literal Definitions    3-Jan-1985 17:02:39 [DS.CRD.V020]MENTSTQ.B32;1   (7)

;    0068  1 %SBTTL 'Macro - Literal Definitions'
;    0069  1
;    0070  1  $CRD_Literals;
;    0071  1  $MEN_Literals;
```

```
;    0072  1 %SBTTL 'Test Category Table'
;    0073  1
;    0074  1 BIND
;    0075  1     MEN$GA_TstCtgTbl =
;    0076  1   PLIT  (
;    0077  1   MEN$K_TstCtg_CPU,
;    0078  1   MEN$K_TstCtg_Memory,
;    0079  1   MEN$K_TstCtg_IO_Adaptor,
;    0080  1   MEN$K_TstCtg_IO_Controller,
;    0081  1   MEN$K_TstCtg_IO_Unit
;    0082  1   ) : VECTOR [, LONG],
;    0083  1
;    0084  1     MEN$GL_TstCtgTbl_Entries = (MEN$GA_TstCtgTbl - 4);
;    0085  1
```

```
;    0086  1 %SBTTL 'MEN$BldTstQ_FncTst_All Routine'
;    0087  1
;    0088  1 GLOBAL ROUTINE MEN$BldTstQ_FncTst_All =
;    0089  1
;    0090  1 !++
;    0091  1 ! FUNCTIONAL DESCRIPTION:
;    0092  1 !
;    0093  1 !    Builds a Test Queue DDB pointer Table for all Supported
;    0094  1 !    hardware associated with CRD MENU Functional Testing.
;    0095  1 !    The order of the DDB pointers starting at the first
;    0096  1 !    DDB pointer entry is the order in which device testing is
;    0097  1 !    expected to execute.
;    0098  1 !
;    0099  1 !    Table entries are entered by searching for DDB's on the basis of
;    0100  1 !    Test Category and Test Class determined by the
;    0101  1 !    Device Test Category table and the Test Menu Table respectively.
;    0102  1 !
;    0103  1 ! INPUT PARAMETERS
;    0104  1 !
;    0105  1 ! OUTPUT PARAMETERS
;    0106  1 !
;    0107  1 ! IMPLICIT INPUTS
;    0108  1 !
;    0109  1 !    DDB Database pointed to by MEN$GA_DDB_Head.
;    0110  1 !
;    0111  1 !    Test Queue table pointed to by MEN$GA_Test_Queue.  It is assumed that
;    0112  1 !    the table consists of a minimum number of longwords equal to the
;    0113  1 !    number of DDB's in the DDB Database plus one(1).
;    0114  1 !
;    0115  1 !    Device Test Category Table at MEN$GA_TstCtgTbl.
;    0116  1 !
;    0117  1 !    Test Menu Table at MEN$GA_TMTbl specifying Device Test Class.
;    0118  1 !
;    0119  1 ! EXPLICIT OUTPUTS
;    0120  1 !
;    0121  1 !    A DDB will only be entered once in the table.
;    0122  1 !
;    0123  1 !    The Table entries are valid until the first null entry.
;    0124  1 !
;    0125  1 !    The last entry of the Table will always be a null.
;    0126  1 !
;    0127  1 ! SIDE EFFECTS
;    0128  1 !
;    0129  1 !    Test Queue Table is initialized with nulls at the start of the routine
;    0130  1 !
;    0131  1 ! COMPLETION CODES
;    0132  1 !
;    0133  1 !    CRD$K_Success  DDB pointers have been entered (Note: If there are
;    0134  1 !     more DDB's to be entered than available slots this
;    0135  1 !    routine will return with CRD$K_Success having filled
;    0136  1 !   all available slots).
;    0137  1 !--
```

```
;    0138  2 BEGIN
;    0139  2     REGISTER
;    0140  2  TstCla,
;    0141  2  TstCtg,
;    0142  2  TstQ_Tbl_Index,
;    0143  2  TstQ_Tbl_Entries,
;    0144  2  TstQ_Tbl_Ptr  : REF VECTOR [, LONG],
;    0145  2  TstCtg_Tbl_Index,
;    0146  2  TMTbl_Index,
;    0147  2  Queue_BlkPtr  : REF $MEN_Queue_Link_ATTR,
;    0148  2  DDB_Ptr  : REF Device_Data_Block_Structure,
;    0149  2  SupBlk_Ptr  : REF $MEN_SupBlk_Attr;
;    0150  2
;    0151  2     !+
;    0152  2     ! Initialize the Test Queue Table with nulls
;    0153  2     !-
;    0154  2
;    0155  2     CH$FILL (0, .MEN$GA_Test_Queue [MEN$V_TQ_Table_Size], .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr]);
;    0156  2
;    0157  2     TstQ_Tbl_Index   = 0;
;    0158  2     TstQ_Tbl_Entries = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Numb_Entries];
;    0159  2     TstQ_Tbl_Ptr     = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr];
;    0160  2
;    0161  2     !+
;    0162  2     ! Get a test category until Test Category Table is ended
;    0163  2     !-
;    0164  2
;    0165  2     TstCtg_Tbl_Index = -1;
;    0166  2
;    0167  2     WHILE ((TstCtg_Tbl_Index = .TstCtg_Tbl_Index + 1) LSS .MEN$GL_TstCtgTbl_Entries) DO
;    0168  3  BEGIN
;    0169  3  TstCtg = .MEN$GA_TstCtgTbl [.TstCtg_Tbl_Index];
;    0170  3
;    0171  3  !+
;    0172  3  ! Get a test class until the Test Menu Table is ended
;    0173  3  !-
;    0174  3
;    0175  3  TMTbl_Index = -1;
;    0176  3
;    0177  3  WHILE ((TMTbl_Index = .TMTbl_Index + 1) LSS MEN$GK_TMTbl_Entries) DO
;    0178  4      BEGIN
;    0179  4      TstCla = .MEN$GA_TMTbl [.TMTbl_Index, TMTbl$B_Entry_TstCla];
;    0180  4
;    0181  4      !+
;    0182  4      ! Find all devices (DDB's) which are supported and represent
;    0183  4      ! the Test Category and Test Class specified and enter into the
;    0184  4      ! the Test Queue Table.  Always leave the last entry of the table null.
;    0185  4      ! If the last entry is reached abort the routine.
;    0186  4      !-
;    0187  4
;    0188  4      Queue_BlkPtr = MEN$GA_DDB Head;
;    0189  4
;    0190  5      WHILE (((Queue_BlkPtr = .Queue_BlkPtr [Queue$l_Flink]) NEQ MEN$GA_DDB_Head) AND
;    0191  4       (.TstQ_Tbl_Index LSS (.TstQ_Tbl_Entries - 1)))  DO
;    0192  5   BEGIN
```

```
;    0193  5    DDB_Ptr = .Queue_BlkPtr;
;    0194  5
;    0195  5     IF (SupBlk_Ptr = .DDB_Ptr [CRD$K_DDB_SupBlk_Block_Ptr]) NEQ 0
;    0196  5    THEN
;    0197  5        IF (.SupBlk_Ptr [SupBlk$B_Test_Category] EQL .TstCtg) AND
;    0198  6           (.SupBlk_Ptr [SupBlk$B_Test_Class]    EQL .TstCla)
;    0199  5        THEN
;    0200  6            BEGIN
;    0201  6     TstQ_Tbl_Ptr [.TstQ_Tbl_Index] = .DDB_Ptr;
;    0202  6     TstQ_Tbl_Index = .TstQ_Tbl_Index + 1;
;    0203  5            END;
;    0204  4    END;  ! WHILE ... DO ...
;    0205  3       END;  ! WHILE ... DO ...
;    0206  2   END;   ! WHILE ... DO ...
;    0207  2      RETURN (CRD$K_Success);
;    0208  1 END;


          .TITLE  MENTSTQ CRD MENU - Test Queue Tables
          .IDENT  \01-00\

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      51  54  53  54  4E  45  4D  07  00000 P.AAA:  .ASCII  <7>\MENTSTQ\          ;
       00000005  00008     .LONG   5              ;
  00000005  00000004  00000003  00000002  00000001  0000C P.AAB:  .LONG   1, 2, 3, 4, 5          ;

          $MODULE-       P.AAA
          MEN$GA_TSTCTGTBL=   P.AAB
          MEN$GL_TSTCTGTBL_ENTRIES=
                P.AAB-4
          .EXTRN  MEN$BLDTSTQ_FNCTST_ALL
          .EXTRN  MEN$BLDTSTQ_FNCTST_DEV
          .EXTRN  MEN$GA_TEST_QUEUE
          .EXTRN  MEN$GA_TMTBL, MEN$GA_DDB_HEAD

          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            07FC 00000     .ENTRY  MEN$BLDTSTQ_FNCTST_ALL, Save R2,R3,R4,R5,-  ; 0088
           R6,R7,R8,R9,R10         ;
00000000G  EF          00        6E      00  2C 00002     MOVC5   #0, (SP), #0, MEN$GA_TEST_QUEUE+8, -       ; 0155
       00000000G  FF      0000B     @MEN$GA_TEST_QUEUE+4          ;
           52  D4 00010     CLRL    TSTQ_TBL_INDEX          ; 0157
           53 00000000G  EF  7D 00012     MOVQ    MEN$GA_TEST_QUEUE, TSTQ_TBL_ENTRIES       ; 0158
           55       01  CE 00019     MNEGL   #1, TSTCTG_TBL_INDEX          ; 0165
           5E  11 0001C     BRB     6$              ; 0167
           51 00000000'EF45  D0 0001E 1$:   MOVL    MEN$GA_TSTCTGTBL[TSTCTG_TBL_INDEX], TSTCTG  ; 0169
           56       01  CE 00026     MNEGL   #1, TMTBL_INDEX          ; 0175
           4A  11 00029     BRB     5$              ; 0177
       00000000GEF46  3F 0002B 2$:   PUSHAW  MEN$GA_TMTBL+1[TMTBL_INDEX]         ; 0179
           50       9E  9A 00032     MOVZBL  @(SP)+, TSTCLA          ;
           57 00000000G  EF  9E 00035     MOVAB   MEN$GA_DDB_HEAD, QUEUE_BLKPTR         ; 0188
           1F  11 0003C     BRB     4$              ; 0190
           58       57  D0 0003E 3$:   MOVL    QUEUE_BLKPTR, DDB_PTR          ; 0193
           59 20    A8  D0 00041     MOVL    32(DDB_PTR), SUPBLK_PTR         ; 0195
```

```
            16  13 00045     BEQL    4$              ;
    51      10   A9          08      00  ED 00047    CMPZV   #0, #8, 16(SUPBLK_PTR), TSTCTG      ; 0197
            0E  12 0004D     BNEQ    4$              ;
    50      11   A9          08      00  ED 0004F    CMPZV   #0, #8, 17(SUPBLK_PTR), TSTCLA      ; 0198
            06  12 00055     BNEQ    4$              ;
          6442       58  D0 00057     MOVL    DDB_PTR, (TSTQ_TBL_PTR)[TSTQ_TBL_INDEX]     ; 0201
            52  D6 0005B     INCL    TSTQ_TBL_INDEX          ; 0202
            57       67  D0 0005D 4$:    MOVL    (QUEUE_BLKPTR), QUEUE_BLKPTR        ; 0190
            5A 00000000G  EF  9E 00060     MOVAB   MEN$GA_DDB_HEAD, R10        ;
            5A       57  D1 00067     CMPL    QUEUE_BLKPTR, R10          ;
            09  13 0006A     BEQL    5$              ;
            5A FF    A3  9E 0006C     MOVAB   -1(R3), R10         ; 0191
            5A       52  D1 00070     CMPL    TSTQ_TBL_INDEX, R10         ;
            C9  19 00073     BLSS    3$              ;
            56  D6 00075 5$:    INCL    TMTBL_INDEX         ; 0177
            0B       56  D1 00077     CMPL    TMTBL_INDEX, #11           ;
            AF  19 0007A     BLSS    2$              ;
            55  D6 0007C 6$:    INCL    TSTCTG_TBL_INDEX        ; 0167
       00000000'  EF       55  D1 0007E     CMPL    TSTCTG_TBL_INDEX, MEN$GL_TSTCTGTBL_ENTRIES  ;
            97  19 00085     BLSS    1$              ;
            50       01  D0 00087     MOVL    #1, R0          ; 0207
         04 0008A     RET                 ; 0208

; Routine Size:  139 bytes,    Routine Base:  CODE + 0000
```

```
;    0209  1
;    0210  1 %SBTTL 'MEN$BldTstQ_FncTst_Dev Routine'
;    0211  1
;    0212  1 GLOBAL ROUTINE MEN$BldTstQ_FncTst_Dev ( I_DDB ) =
;    0213  1
;    0214  1 !++
;    0215  1 ! FUNCTIONAL DESCRIPTION:
;    0216  1 !
;    0217  1 !    Builds a Test Queue DDB pointer Table for the supported
;    0218  1 !    hardware device specified by the DDB and associated
;    0219  1 !    with CRD MENU Functional Testing.
;    0220  1 !
;    0221  1 !
;    0222  1 ! INPUT PARAMETERS
;    0223  1 !
;    0224  1 !    I_DDB  Address of DDB
;    0225  1 !
;    0226  1 ! OUTPUT PARAMETERS
;    0227  1 !
;    0228  1 ! IMPLICIT INPUTS
;    0229  1 !
;    0230  1 !    Test Queue table pointed to by MEN$GA_Test_Queue.  It is assumed that
;    0231  1 !    the table consists of a minimum of two(2) longwords .
;    0232  1 !
;    0233  1 ! EXPLICIT OUTPUTS
;    0234  1 !
;    0235  1 !    Only one table entry is entered.
;    0236  1 !
;    0237  1 ! SIDE EFFECTS
;    0238  1 !
;    0239  1 !    Test Queue Table is nitialized with nulls at the beginning of the routine
;    0240  1 !
;    0241  1 ! COMPLETION CODES
;    0242  1 !
;    0243  1 !    CRD$K_Success
;    0244  1 !--
```

```
;    0245  2 BEGIN
;    0246  2     LOCAL
;    0247  2  TstQ_Tbl_Ptr : REF VECTOR [, LONG];
;    0248  2
;    0249  2     BIND
;    0250  2  DDB = .I_DDB;
;    0251  2
;    0252  2     CH$FILL (0, .MEN$GA_Test_Queue [MEN$V_TQ_Table_Size], .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr]);
;    0253  2      ! Fill Table with nulls
;    0254  2
;    0255  2     TstQ_TBl_Ptr = .MEN$GA_Test_Queue [MEN$V_TQ_Table_Ptr];
;    0256  2      ! Point to Hardware Support Table
;    0257  2
;    0258  2     TstQ_Tbl_Ptr [0] = DDB;  ! Insert DDB pointer
;    0259  2
;    0260  2     RETURN (CRD$K_Success);
;    0261  1 END;



            003C 00000     .ENTRY  MEN$BLDTSTQ_FNCTST_DEV, Save R2,R3,R4,R5    ; 0212
00000000G   EF          00        6E      00   2C 00002    MOVC5   #0, (SP), #0, MEN$GA_TEST_QUEUE+8, -       ; 0252
       00000000G   FF      0000B    @MEN$GA_TEST_QUEUE+4        ;
           50 00000000G   EF   D0 00010    MOVL    MEN$GA_TEST_QUEUE+4, TSTQ_TBL_PTR       ; 0255
           60 04    AC   D0 00017    MOVL    I_DDB, (TSTQ_TBL_PTR)          ; 0258
           50       01   D0 0001B    MOVL    #1, R0          ; 0260
        04 0001E    RET                 ; 0261

; Routine Size:  31 bytes,    Routine Base:  CODE + 008B
```

```
;   0262  1 END
;   0263  0 ELUDOM



;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA           32  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          170  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



Symbol                            Type  Defined  Referenced ...
--------------  ---- -------- --      - --- - ---------------
$CRD_LITERALS   Macro    Lib03    70
$EQULST    Macro    Lib04    70       71
$ER    Comptime    67
$MEN_LITERALS   Macro    Lib03    71
$MEN_QUEUE_LINK_ATTR  Macro    Lib03   147
$MEN_QUEUE_LINK_FLD  Fieldset Lib03   147
$MEN_SUPBLK_ATTR  Macro    Lib03   149
$MEN_SUPBLK_FLD   Fieldset Lib03   149
$MODULE    Bind         67
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     70
AUTO$K_EXIT_CONTROL_C  Literal     70
AUTO$K_EXIT_DUMMY_2  Literal     70
AUTO$K_EXIT_DUMMY_3  Literal     70
AUTO$K_EXIT_ERRORS_COMPLETE Literal     70
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     70
AUTO$K_EXIT_INTERNAL_ERROR Literal     70
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     70
AUTO$K_EXIT_LAST_REASON  Literal     70     70
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     70
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     70
AUTO$K_EXIT_NOERRORS_PREP Literal     70
CRD$K_ACTION_EQL_DO_NOTHING Literal     70
CRD$K_ACTION_RANGE_ERROR Literal     70
CRD$K_ADVANCE_DIAGNOSTIC Literal     70
CRD$K_ADVANCE_GENERAL_COM Literal     70
CRD$K_ADVANCE_STATE  Literal     70
CRD$K_ALLOCATION_ERROR  Literal     70
CRD$K_ASSIGN_PTABLE_PTRS Literal     70
CRD$K_AUTOSIZER_ERROR  Literal     70
CRD$K_AUTOSIZER_SET_FLAGS Literal     70
CRD$K_BAD_ACTION_NUMBER  Literal     70
CRD$K_BAD_TYPECODE_ERROR Literal     70
CRD$K_BASE_SYSTEM_IDC  Literal     70
CRD$K_BASE_SYSTEM_LESI  Literal     70
```

Symbol	Type	Defined	Referenced ...
CRD$K_BASE_SYSTEM_NULL	Literal	70	
CRD$K_BASE_SYSTEM_UDA_0	Literal	70	
CRD$K_BASE_SYSTEM_UDA_1	Literal	70	
CRD$K_BASE_SYSTEM_UDA_2	Literal	70	
CRD$K_BASE_SYSTEM_UDA_3	Literal	70	
CRD$K_CRD_INITIALIZATION	Literal	70	
CRD$K_CREATE_HST	Literal	70	
CRD$K_DATA_LENGTH_ERROR	Literal	70	
CRD$K_DDB_AUTO_SUPPORT_ENTRY	Literal	70	
CRD$K_DDB_BLINK	Literal	70	
CRD$K_DDB_FLINK	Literal	70	
CRD$K_DDB_LAST_ENTRY	Literal	70	
CRD$K_DDB_MEMORY_SIZE	Literal	70	
CRD$K_DDB_MENU_DEVICE_NUMB	Literal	70	
CRD$K_DDB_MENU_SUPPORT_ENTRY	Literal	70	
CRD$K_DDB_MENU_SUPPORT_SIZE	Literal	70	
CRD$K_DDB_NUMB_SUPPORT_TABLES	Literal	70	
CRD$K_DDB_PTABLE_ENTRY	Literal	70	
CRD$K_DDB_STATUS	Literal	70	
CRD$K_DDB_SUPBLK_BLOCK_PTR	Literal	70	195
CRD$K_DEVICE_NAME	Literal	70	
CRD$K_DIAGNOSTIC_ERROR	Literal	70	
CRD$K_DIAGNOSTIC_NAME	Literal	70	
CRD$K_DONE_GENERAL_COMS	Literal	70	
CRD$K_DO_NOTHING	Literal	70	
CRD$K_DUMMY_10	Literal	70	
CRD$K_DUMMY_11	Literal	70	
CRD$K_DUMMY_12	Literal	70	
CRD$K_DUMMY_13	Literal	70	
CRD$K_DUMMY_3	Literal	70	
CRD$K_DUMMY_4	Literal	70	
CRD$K_DUMMY_5	Literal	70	
CRD$K_DUMMY_6	Literal	70	
CRD$K_DUMMY_7	Literal	70	
CRD$K_DUMMY_8	Literal	70	
CRD$K_DUMMY_9	Literal	70	
CRD$K_EXIT_CRD	Literal	70	
CRD$K_HOOK_POINT	Literal	70	
CRD$K_HS_INTERNAL_ERROR	Literal	70	
CRD$K_IDC_EVDLB	Literal	70	
CRD$K_IDC_EVDLC	Literal	70	
CRD$K_IDC_EVDLD	Literal	70	
CRD$K_IDC_EVRAD_0	Literal	70	
CRD$K_IDC_EVRAD_1	Literal	70	
CRD$K_IDC_LAST_DIAGNOSTIC	Literal	70	
CRD$K_INTERNAL_ERROR	Litera	70	
CRD$K_LAST_ACTION_ROUTINE	Literal	70	
CRD$K_LAST_ENTRY	Literal	70	
CRD$K_LAST_FUNCTION	Literal	70	
CRD$K_LAST_HOOK_POINT	Literal	70	
CRD$K_LAST_INTERNAL_ERROR	Literal	70	
CRD$K_LAST_TYPE	Literal	70	
CRD$K_LESI_ENWCA	Literal	70	

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_LESI_EVRMB_0  Literal     70
CRD$K_LESI_EVRMB_1  Literal     70
CRD$K_LESI_LAST_DIAGNOSTIC Literal     70
CRD$K_LEVEL_4_PASSED  Literal     70
CRD$K_LOAD_AUTOSIZER  Literal     70
CRD$K_LOAD_ERROR  Literal     70
CRD$K_LOAD_GENERAL_COM  Literal     70
CRD$K_LOAD_LOAD_COM  Literal     70
CRD$K_LOAD_SELECT_COM  Literal     70
CRD$K_LOAD_SET_EVENT_COM Literal     70
CRD$K_LOAD_SET_FLAGS_COM Literal     70
CRD$K_LOAD_START_COM  Literal     70
CRD$K_LOAD_SUP_FILE  Literal     70
CRD$K_MM_INTERNAL_ERROR  Literal     70
CRD$K_NEW_LINE   Literal     70
CRD$K_PREPARATION_ERROR  Literal     70
CRD$K_PREPARATION_ERROR_TEXT Literal     70
CRD$K_RUNTIME   Literal     70
CRD$K_SAME_LINE   Literal     70
CRD$K_SET_EVENT_ARGS  Literal     70
CRD$K_SET_FLAGS_ARGS  Literal     70
CRD$K_SET_UP_DIAGNOSTIC  Literal     70
CRD$K_START   Literal     70
CRD$K_START_AUTOSIZER  Literal     70
CRD$K_START_ERROR  Literal     70
CRD$K_START_QUALIFIERS  Literal     70
CRD$K_STAR_LINE   Literal     70
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal     70
CRD$K_STATE_ADVANCE_GENERAL_COM Literal     70
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal     70
CRD$K_STATE_AUTOSIZER_ERROR Literal     70
CRD$K_STATE_AUTOSIZER_RUNNING Literal     70
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal     70
CRD$K_STATE_DIAGNOSTIC_ERROR Literal     70
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal     70
CRD$K_STATE_DONE_DIAGNOSTICS Literal     70
CRD$K_STATE_DONE_GENERAL_COMS Literal     70
CRD$K_STATE_DUMMY_1  Literal     70
CRD$K_STATE_DUMMY_10  Literal     70
CRD$K_STATE_DUMMY_12  Literal     70
CRD$K_STATE_DUMMY_13  Literal     70
CRD$K_STATE_DUMMY_2  Literal     70
CRD$K_STATE_DUMMY_3  Literal     70
CRD$K_STATE_DUMMY_4  Literal     70
CRD$K_STATE_DUMMY_6  Literal     70
CRD$K_STATE_DUMMY_7  Literal     70
CRD$K_STATE_DUMMY_8  Literal     70
CRD$K_STATE_EXIT_CRD  Literal     70
CRD$K_STATE_INTERNAL_ERROR Literal     70
CRD$K_STATE_LAST_STATE  Literal     70
CRD$K_STATE_LEVEL_4_PASSED Literal     70
CRD$K_STATE_LOAD_AUTOSIZER Literal     70
CRD$K_STATE_LOAD_ERROR  Literal     70
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------- -------- ----- ---------------
CRD$K_STATE_LOAD_GENERAL_COM Literal      70
CRD$K_STATE_LOAD_LOAD_COM Literal     70
CRD$K_STATE_LOAD_SELECT_COM Literal      70
CRD$K_STATE_LOAD_SET_EVENT_COM Literal      70
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal      70
CRD$K_STATE_LOAD_START_COM Literal     70
CRD$K_STATE_PREPARATION_ERROR Literal      70
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal      70
CRD$K_STATE_START  Literal     70
CRD$K_STATE_START_AUTOSIZER Literal      70
CRD$K_STATE_START_ERROR  Literal     70
CRD$K_SUCCESS   Literal  Lib03   207      260
CRD$K_TEST_START_TEXT  Literal     70
CRD$K_TQ_INTERNAL_ERROR  Literal      70
CRD$K_TYPECODE   Literal     70
CRD$K_UDA_0_EVDLB  Literal      70
CRD$K_UDA_0_EVDLC  Literal      70
CRD$K_UDA_0_EVDLD  Literal      70
CRD$K_UDA_0_EVRLA_0  Literal      70
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal      70
CRD$K_UDA_1_EVDLB  Literal      70
CRD$K_UDA_1_EVDLC  Literal      70
CRD$K_UDA_1_EVDLD  Literal      70
CRD$K_UDA_1_EVRLA_0  Literal      70
CRD$K_UDA_1_EVRLA_1  Literal      70
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal      70
CRD$K_UDA_2_EVDLB  Literal      70
CRD$K_UDA_2_EVDLC  Literal      70
CRD$K_UDA_2_EVDLD  Literal      70
CRD$K_UDA_2_EVRLA_0  Literal      70
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal      70
CRD$K_UDA_3_EVDLB  Literal      70
CRD$K_UDA_3_EVDLC  Literal      70
CRD$K_UDA_3_EVDLD  Literal      70
CRD$K_UDA_3_EVRLA_0  Literal      70
CRD$K_UDA_3_EVRLA_1  Literal      70
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal      70
DDB    Bind        250   258a
DDB_PTR    Register   148   193=    195a.   201.
DEVICE_DATA_BLOCK_STRUCTURE Structur Lib03   148
GET1ST_    Macro    Lib04    70       71
GET2ND_    Unbound            70       71
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal      70
HS$K_ACTION_ADVANCE_GENERAL_COM Literal      70
HS$K_ACTION_ADVANCE_STATE Literal      70
HS$K_ACTION_CHANGE_TABLE Literal      70
HS$K_ACTION_DIAGNOSTIC_ERROR Literal      70
HS$K_ACTION_DONE_GENERAL_COMS Literal      70
HS$K_ACTION_DO_NOTHING  Literal      70
HS$K_ACTION_DUMMY_3  Literal      70
HS$K_ACTION_DUMMY_4  Literal      70
HS$K_ACTION_DUMMY_5  Literal      70
HS$K_ACTION_DUMMY_6  Literal      70
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_ACTION_INIT_HS  Literal     70
HS$K_ACTION_INTERNAL_ERROR Literal     70
HS$K_ACTION_LAST_ACTION  Literal     70
HS$K_ACTION_LOAD_ERROR  Literal     70
HS$K_ACTION_LOAD_GENERAL_COM Literal     70
HS$K_ACTION_LOAD_LOAD_COM Literal     70
HS$K_ACTION_LOAD_SELECT_COM Literal     70
HS$K_ACTION_LOAD_SET_EVENT_COM Literal     70
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal     70
HS$K_ACTION_LOAD_START_COM Literal     70
HS$K_ACTION_PREPARATION_ERROR Literal     70
HS$K_ACTION_START_ERROR  Literal     70
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal     70
HS$K_STATE_ADVANCE_GENERAL_COM Literal     70
HS$K_STATE_CHANGE_TABLE  Literal     70
HS$K_STATE_DIAGNOSTIC_ERROR Literal     70
HS$K_STATE_DIAGNOSTIC_RUNNING Literal     70
HS$K_STATE_DONE_GENERAL_COMS Literal     70
HS$K_STATE_DUMMY_1  Literal     70
HS$K_STATE_DUMMY_2  Literal     70
HS$K_STATE_DUMMY_3  Literal     70
HS$K_STATE_DUMMY_4  Literal     70
HS$K_STATE_DUMMY_6  Literal     70
HS$K_STATE_INIT_HS  Literal     70
HS$K_STATE_INTERNAL_ERROR Literal     70
HS$K_STATE_LAST_STATE  Literal     70
HS$K_STATE_LOAD_ERROR  Literal     70
HS$K_STATE_LOAD_GENERAL_COM Literal     70
HS$K_STATE_LOAD_LOAD_COM Literal     70
HS$K_STATE_LOAD_SELECT_COM Literal     70
HS$K_STATE_LOAD_SET_EVENT_COM Literal     70
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     70
HS$K_STATE_LOAD_START_COM Literal     70
HS$K_STATE_PREPARATION_ERROR Literal     70
HS$K_STATE_START_ERROR  Literal     70
I_DDB    RoutForm   212   250.
MEN$BLDTSTQ_FNCTST_ALL  GlobRout    88 Lib03e     53f
MEN$BLDTSTQ_FNCTST_DEV  GlobRout   212 Lib03e     54f
MEN$GA_DDB_HEAD   External Lib03   188a    190
MEN$GA_TEST_QUEUE  External Lib03   155.    155a=   158.    159.    252.    252a=   255.
MEN$GA_TMTBL   External Lib03   179.
MEN$GA_TSTCTGTBL  Bind        75    84a    169.
MEN$GK_TEST_QUEUE  Literal     71
MEN$GK_TMTBL_ENTRIES  Literal  Lib03   177
MEN$GL_TSTCTGTBL_ENTRIES Bind         84   167.
MEN$K_CNTLC_SEL_CONTINUE Literal     71
MEN$K_CNTLC_SEL_EXIT  Literal     71
MEN$K_CNTLC_SEL_FTMENU  Literal     71
MEN$K_CNTLC_SEL_LAST_ENTRY Literal     71
MEN$K_DEVTSTPREP_1  Literal     71
MEN$K_DEVTSTPREP_2  Literal     71
MEN$K_DEVTSTPREP_3  Literal     71
MEN$K_DEVTSTPREP_4  Literal     71
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MEN$K_DEVTSTPREP_5  Literal     71
MEN$K_DEVTSTPREP_6  Literal     71
MEN$K_DEVTSTPREP_7  Literal     71
MEN$K_DEVTSTPREP_NULL  Literal     71
MEN$K_FTMRET_EXIT  Literal     71
MEN$K_FTMRET_FAILURE  Literal     71
MEN$K_FTMRET_MENU  Literal     71
MEN$K_FTMRET_TEST  Literal     71
MEN$K_FTMSEL_EXIT  Literal     71
MEN$K_FTMSEL_FNCTST_ALL  Literal     71
MEN$K_FTMSEL_LAST_ENTRY  Literal     71
MEN$K_FTMSEL_PREP  Literal     71
MEN$K_FTMSEL_SYSTEM_HDWR Literal     71
MEN$K_FTMSEL_TEST  Literal     71
MEN$K_HS_STATE_TABLE  Literal     70
MEN$K_MM_STATE_TABLE  Literal     70
MEN$K_PREP_ERROR_TEXT_LEN Literal     71
MEN$K_QUEUE_LINK_SIZE  Literal  Lib03   147
MEN$K_SELDEV_NUMB_START  Literal     71
MEN$K_SUPBLK_SIZE  Literal  Lib03   149
MEN$K_TMENU_TSTALL_DEVNUMB Literal     71
MEN$K_TQ_STATE_TABLE  Literal     70
MEN$K_TSTCLA_ALL  Literal     71
MEN$K_TSTCLA_CARD  Literal     71
MEN$K_TSTCLA_COMM  Literal     71
MEN$K_TSTCLA_CPU  Literal     71
MEN$K_TSTCLA_DISK  Literal     71
MEN$K_TSTCLA_IO_ADAPTOR  Literal     71
MEN$K_TSTCLA_LAST_ENTRY  Literal     71
MEN$K_TSTCLA_MEMORY  Literal     71
MEN$K_TSTCLA_NULL  Literal     71
MEN$K_TSTCLA_PRINTER  Literal     71
MEN$K_TSTCLA_REAL  Literal     71
MEN$K_TSTCLA_TAPE  Literal     71
MEN$K_TSTCLA_TERM  Literal     71
MEN$K_TSTCTG_CPU  Literal     71    77
MEN$K_TSTCTG_IO_ADAPTOR  Literal     71    79
MEN$K_TSTCTG_IO_CONTROLLER Literal     71    80
MEN$K_TSTCTG_IO_UNIT  Literal     71    81
MEN$K_TSTCTG_MEMORY  Literal     71    78
MEN$K_TSTCTG_NULL  Literal     71
MEN$K_TSTTXT_DEVNAM_SZ  Literal     71
MEN$K_TSTTXT_TSTCLA_SZ  Literal     71
MEN$K_TSTTXT_UUTNAM_SZ  Literal     71
MEN$V_TQ_TABLE_NUMB_ENTRIES Field    Lib03   158.
MEN$V_TQ_TABLE_PTR  Field    Lib03   155a=   159.    252a=   255.
MEN$V_TQ_TABLE_SIZE  Field    Lib03   155.    252.
MENU$K_EXIT_AUTOSIZER_ERROR Literal     70
MENU$K_EXIT_INTERNAL_ERROR Literal     70
MENU$K_EXIT_LAST_REASON  Literal     70
MENU$K_EXIT_OPERATOR_ABORT Literal     70
MENU$K_EXIT_OPERATOR_STOP Literal     70
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     70
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     70
MM$K_ACTION_ADVANCE_STATE Literal      70
MM$K_ACTION_AUTOSIZER_ERROR Literal      70
MM$K_ACTION_BUILD_DDBS  Literal      70
MM$K_ACTION_BUILD_HSTS  Literal      70
MM$K_ACTION_CHANGE_TABLE Literal      70
MM$K_ACTION_DONE_INITS  Literal      70
MM$K_ACTION_DO_NOTHING  Literal      70
MM$K_ACTION_DUMMY_3  Literal      70
MM$K_ACTION_DUMMY_4  Literal      70
MM$K_ACTION_DUMMY_5  Literal      70
MM$K_ACTION_DUMMY_6  Literal      70
MM$K_ACTION_DUMMY_7  Literal      70
MM$K_ACTION_DUMMY_8  Literal      70
MM$K_ACTION_INTERNAL_ERROR Literal      70
MM$K_ACTION_LAST_ACTION  Literal      70
MM$K_ACTION_LOAD_AUTOSIZER Literal      70
MM$K_ACTION_MENU_PROMPT  Literal      70
MM$K_ACTION_PRINT_MENU  Literal      70
MM$K_ACTION_PRINT_MENU_HEADING Literal      70
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal      70
MM$K_ACTION_START_AUTOSIZER Literal      70
MM$K_STATE_AUTOSIZER_ERROR Literal      70
MM$K_STATE_AUTOSIZER_RUNNING Literal      70
MM$K_STATE_BUILD_DDBS  Literal      70
MM$K_STATE_BUILD_HSTS  Literal      70
MM$K_STATE_CHANGE_TABLE  Literal      70
MM$K_STATE_DONE_INITS  Literal      70
MM$K_STATE_DUMMY_1  Literal      70
MM$K_STATE_DUMMY_2  Literal      70
MM$K_STATE_DUMMY_3  Literal      70
MM$K_STATE_DUMMY_5  Literal      70
MM$K_STATE_DUMMY_7  Literal      70
MM$K_STATE_DUMMY_8  Literal      70
MM$K_STATE_INTERNAL_ERROR Literal      70
MM$K_STATE_LAST_STATE  Literal      70
MM$K_STATE_LOAD_AUTOSIZER Literal      70
MM$K_STATE_MENU_PROMPT  Literal      70
MM$K_STATE_PRINT_MENU  Literal      70
MM$K_STATE_PRINT_MENU_HEADING Literal      70
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal      70
MM$K_STATE_START  Literal      70
MM$K_STATE_START_AUTOSIZER Literal      70
MODNAM    Macro    Lib01    67
NUL2ND_    Macro    Lib04    70       71
QUEUE$L_FLINK   Field    Lib03   190.
QUEUE_BLKPTR   Register   147   188=    190=    190a.   193.
SUPBLK$B_TEST_CATEGORY  Field    Lib03   197.
SUPBLK$B_TEST_CLASS  Field    Lib03   198.
SUPBLK_PTR   Register   149   195=    197a.   198a.
TMTBL$B_ENTRY_TSTCLA  Field    Lib03   179.
TMTBL_INDEX   Register   146   175=    177=    177.    179.
TQ$K_ACTION_ADVANCE_STATE Literal      70
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
TQ$K_ACTION_CHANGE_TABLE Literal     70
TQ$K_ACTION_DONE_TEST_QUEUE Literal      70
TQ$K_ACTION_DO_NOTHING  Literal      70
TQ$K_ACTION_DUMMY_3  Literal      70
TQ$K_ACTION_DUMMY_4  Literal      70
TQ$K_ACTION_DUMMY_5  Literal      70
TQ$K_ACTION_GET_DDB  Literal      70
TQ$K_ACTION_INIT_TQ  Literal      70
TQ$K_ACTION_INTERNAL_ERROR Literal      70
TQ$K_ACTION_LAST_ACTION  Literal      70
TQ$K_STATE_CHANGE_TABLE  Literal      70
TQ$K_STATE_DONE_TEST_QUEUE Literal      70
TQ$K_STATE_DUMMY_1  Literal      70
TQ$K_STATE_DUMMY_2  Literal      70
TQ$K_STATE_DUMMY_3  Literal      70
TQ$K_STATE_DUMMY_4  Literal      70
TQ$K_STATE_DUMMY_5  Literal      70
TQ$K_STATE_GET_DDB  Literal      70
TQ$K_STATE_INIT_TQ  Literal      70
TQ$K_STATE_INTERNAL_ERROR Literal      70
TQ$K_STATE_LAST_STATE  Literal      70
TSTCLA     Register   140   179=    198.
TSTCTG     Register   141   169=    197.
TSTCTG_TBL_INDEX  Register   145   165=    167=    167.    169.
TSTQ_TBL_ENTRIES  Register   143   158=    191.
TSTQ_TBL_INDEX   Register   142   157=    191.    201.    202=    202.
TSTQ_TBL_PTR   Register   144   159=    201a=
    Local      247   255=    258a=


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  ---------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]MENTSTQ.B32;1
    2  Module MENTSTQ
   46  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   47  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   48  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   49  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  263  Eludom MENTSTQ
```

```
 KEY TO REFERENCE TYPE FLAGS

      . Fetch
      = Store
      c Routine call
      a Address use
      @ Indirect use
      e External, external routine, or external literal declaration
      f Forward or forward routine declaration
      h Condition handler enabling
      m Map declaration
      u Undeclare declaration


;     Library Statistics
;
;           -- ----- Symbols --------     Pages Proces_ng
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;            671          1          0          46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;            923          0          0          94          00:00.1
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;            486         34          6          62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619          3           0       1000          00:04.3



;      COMMAND QUALIFIERS

; BLISS MENTSTQ.B32/LIST-MENTSTQ.LIS/CROSS_REFERENCE/DEBUG/OBJECT=MENTSTQ.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  170 code + 32 data bytes
; Run Time:     00:20.3
; Elapsed Time:     00:23.1
; Lines/CPU Min:      776
; Lexemes/CPU-Min: 60938
; Memory Used: 178 pages
; Compilation Complete
```

```
;    0001   0 %TITLE '*** MENTURING Module'
;    0002   0 MODULE MENTURING (    ! Routines to access the Menu Test State Tables
;    0003   0    IDENT = 'V01.2'
;    0004   0    ) =
;    0005   0 !++
;    0006   0 ! COPYRIGHT (c) 1980, 1981
;    0007   0 ! DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;    0008   0 !
;    0009   0 ! THIS SOFTWARE IS FURNISHED  UNDER A LICENSE FOR USE ONLY ON A SINGLE
;    0010   0 ! COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
;    0011   0 ! ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
;    0012   0 ! MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;    0013   0 ! EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
;    0014   0 ! TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
;    0015   0 ! REMAIN IN DEC.
;    0016   0 !
;    0017   0 ! THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
;    0018   0 ! AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
;    0019   0 ! CORPORATION.
;    0020   0 !
;    0021   0 ! DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
;    0022   0 ! SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
;    0023   0 !-
```

```
;    0024  0 !++
;    0025  0 ! Facility:
;    0026  0 !
;    0027  0 ! Diagnostic Supervisor (part of the Loadable-CRD image).
;    0028  0 !
;    0029  0 ! Abstract:
;    0030  0 !
;    0031  0 ! This module contains those state routines that will access the three Menu Test state tables.
;    0032  0 !
;    0033  0 ! History:
;    0034  0 !
;    0035  0 ! 01 Jack Stansbury July 30,1982
;    0036  0 !  - Created module
;    0037  0 !
;    0038  0 ! 02 Ron Parker October 18, 1984
;    0039  0 !  - Make Previous state not just used be debug mode.
;    0040  0 !    This is also used by MM$_Start_Autosizer routine
;    0041  0 !    to stay in current state.
;    0042  0 !
;    0043  0 !--
```

```
;    0044  0 %SBTTL 'Libraries'
;    0045  1 BEGIN     ! Begin the MENTURING module
;    0046  1
;    0047  1 LIBRARY
;    0048  1  '$DS';    ! Use Ds.L32 first
;    0049  1
;    0050  1 LIBRARY
;    0051  1  '$Diag';   ! Use Diag.L32 second
;    0052  1
;    0053  1 LIBRARY
;    0054  1  '$CRD';    ! Use CRD.L32 third
;    0055  1
;    0056  1 LIBRARY
;    0057  1  'Sys$Library:Lib';  ! Use Lib as last resort
```

```
;    0058  1 %SBTTL 'Table of Contents'
;    0059  1
;    0060  1 FORWARD ROUTINE
;    0061  1  MEN$Init_MM_State_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0062  1  MEN$Init_TQ_State_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0063  1  MEN$Init_HS_State_Table  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0064  1
;    0065  1  MEN$Get_Next_State  : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0066  1
;    0067  1  MEN$Get_Next_Action_Routine : NOVALUE ADDRESSING_MODE (LONG_RELATIVE),
;    0068  1
;    0069  1  MEN$Turing_Machine  : ADDRESSING_MODE (LONG_RELATIVE),
;    0070  1
;    0071  1  MEN$Print_Action_Routine : NOVALUE ADDRESSING_MODE (LONG_RELATIVE);
;    0072  1
```

```
;    0073  1 %SBTTL 'Included Macros and Literals'
;    0074  1
;    0075  1  $DS_DSADef;   ! Define the DSA locations
;    0076  1  $CRD_Literals;   ! The CRD literals
```

```
;    0077  1 %SBTTL 'Psect Definitions'
;    0078  1
;    0079  1 PSECT
;    0080  1     PLIT    = Data (NOEXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0081  1
;    0082  1 PSECT
;    0083  1     CODE    = Code (  EXECUTE,   SHARE, NOWRITE, ADDRESSING_MODE (LONG_RELATIVE), PIC);
;    0084  1
;    0085  1 PSECT
;    0086  1     OWN     = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
;    0087  1
;    0088  1 PSECT
;    0089  1     GLOBAL  = Work (NOEXECUTE, NOSHARE,   WRITE, ADDRESSING_MODE (LONG_RELATIVE));
```

```
;    0090  1 %SBTTL 'Module-Global Static Storage'
;    0091  1
;    0092  1  ModNam ('MENTURING');  ! Module name for ErrSup macro
```

```
;    0093  1 %SBTTL 'MEN$Init_MM_State_Table Routine'
;    0094  1
     0095  1 GLOBAL ROUTINE MEN$Init_MM_State_Table : NOVALUE =
;    0096  1
;    0097  1 !++
;    0098  1 ! Functional Description:
;    0099  1 !
;    0100  1 ! Initialize the CRD$GAW_MM_State_Table and all its associated data structures.
;    0101  1 !
;    0102  1 ! Formal Input Parameters:
;    0103  1 !
;    0104  1 ! None
;    0105  1 !
;    0106  1 ! Formal Output Parameters:
;    0107  1 !
;    0108  1 ! None
;    0109  1 !
;    0110  1 ! Side Effects:
;    0111  1 !
;    0112  1 ! None
;    0113  1 !
;    0114  1 ! Completion Codes:
;    0115  1 !
;    0116  1 ! None
;    0117  1 !--
```

```
;   0118  2 BEGIN     ! Routine MEN$Init_MM_State_Table
;   0119  2
;   0120  2  CRD$GL_MM_Current_State = MM$K_State_Start;
;   0121  2  RETURN;
;   0122  1 END;     ! Routine MEN$Init_MM_State_Table


          .TITLE  MENTURING *** MENTURING Module
          .IDENT  \V01.2\

          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

      47  4E  49  52  55  54  4E  45  4D  09  00000 P.AAA:  .ASCII  <9>\MENTURING\         ;

          DSA$AL_APTMAIL=     65024
          DSA$GL_FLAGS=       65024
          DSA$GL_APTCOM=      65028
          DSA$GL_PASSES=      65032
          DSA$GL_UNITS=       65036
          DSA$GL_SECTNO=      65040
          DSA$GL_SID=        65044
          DSA$GL_MSGTYP=      65088
          DSA$GL_ERRNO=       65092
          DSA$GL_EVENT=       65096
          DSA$GL_SUBTNO=      65100
          DSA$GL_TESTNO=      65104
          DSA$GL_PASSNO=      65108
          DSA$GL_DEVLEN=      65112
          DSA$GL_DEVNAM=      65116
          DSA$GL_MSGPTR=      65128
          $MODULE=       P.AAA
           .EXTRN  MEN$INIT_MM_STATE_TABLE
           .EXTRN  MEN$INIT_TQ_STATE_TABLE
           .EXTRN  MEN$INIT_HS_STATE_TABLE
           .EXTRN  MEN$TURING_MACHINE
           .EXTRN  MEN$PRINT_ACTION_ROUTINE
           .EXTRN  CRD$GL_MM_CURRENT_STATE

          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            0000 00000     .ENTRY  MEN$INIT_MM_STATE_TABLE, Save nothing       ; 0095
      00000000G  EF      03  D0 00002     MOVL    #3, CRD$GL_MM_CURRENT_STATE        ; 0120
       04 00009     RET                 ; 0122

; Routine Size:  10 bytes,    Routine Base:  CODE + 0000
```

```
;    0123  1 %SBTTL 'MEN$Init_TQ_State_Table Routine'
;    0124  1
;    0125  1 GLOBAL ROUTINE MEN$Init_TQ_State_Table : NOVALUE =
;    0126  1
;    0127  1 !++
;    0128  1 ! Functional Description:
;    0129  1 !
;    0130  1 ! Initialize the CRD$GAW_TQ_State_Table and all its associated data structures.
;    0131  1 !
;    0132  1 ! Formal Input Parameters:
;    0133  1 !
;    0134  1 ! None
;    0135  1 !
;    0136  1 ! Formal Output Parameters:
;    0137  1 !
;    0138  1 ! None
;    0139  1 !
;    0140  1 ! Side Effects:
;    0141  1 !
;    0142  1 ! None
;    0143  1 !
;    0144  1 ! Completion Codes:
;    0145  1 !
;    0146  1 ! None
;    0147  1 !--
```

```
;   0148  2 BEGIN     ! Routine MEN$Init_TQ_State_Table
;   0149  2
;   0150  2  CRD$GL_TQ_Current_State = TQ$K_State_Init_TQ;
;   0151  2  RETURN;
;   0152  1 END;     ! Routine MEN$Init_TQ_State_Table



           .EXTRN  CRD$GL_TQ_CURRENT_STATE

             0000 00000     .ENTRY  MEN$INIT_TQ_STATE_TABLE, Save nothing       ; 0125
       00000000G  EF       03  D0 00002     MOVL    #3, CRD$GL_TQ_CURRENT_STATE         ; 0150
        04 00009     RET                 ; 0152

; Routine Size:  10 bytes,    Routine Base:  CODE + 000A
```

```
;    0153  1 %SBTTL 'MEN$Init_HS_State_Table Routine'
;    0154  1
;    0155  1 GLOBAL ROUTINE MEN$Init_HS_State_Table : NOVALUE =
;    0156  1
;    0157  1 !++
;    0158  1 ! Functional Description:
;    0159  1 !
;    0160  1 ! Initialize the CRD$GAW_HS_State_Table and all its associated data structures.
;    0161  1 !
;    0162  1 ! Formal Input Parameters:
;    0163  1 !
;    0164  1 ! None
;    0165  1 !
;    0166  1 ! Formal Output Parameters:
;    0167  1 !
;    0168  1 ! None
;    0169  1 !
;    0170  1 ! Side Effects:
;    0171  1 !
;    0172  1 ! None
;    0173  1 !
;    0174  1 ! Completion Codes:
;    0175  1 !
;    0176  1 ! None
;    0177  1 !--
```

```
;    0178  2 BEGIN     ! Routine MEN$Init_HS_State_Table
;    0179  2
;    0180  2  CRD$GL_HS_Current_State = HS$K_State_Init_HS;
;    0181  2  RETURN;
;    0182  1 END;     ! Routine MEN$Init_HS_State_Table



           .EXTRN  CRD$GL_HS_CURRENT_STATE

             0000 00000     .ENTRY  MEN$INIT_HS_STATE_TABLE, Save nothing       ; 0155
       00000000G  EF      03  D0 00002     MOVL    #3, CRD$GL_HS_CURRENT_STATE        ; 0180
        04 00009     RET                 ; 0182

; Routine Size:  10 bytes,    Routine Base:  CODE + 0014
```

```
;    0183  1 %SBTTL 'MEN$Get_Next_State Routine'
;    0184  1
;    0185  1 ROUTINE MEN$Get_Next_State : NOVALUE =
;    0186  1
;    0187  1 !++
;    0188  1 ! Functional Description:
;    0189  1 !
;    0190  1 ! Get the next state number from one of the state tables.
;    0191  1 !
;    0192  1 ! Formal Input Parameters:
;    0193  1 !
;    0194  1 ! None
;    0195  1 !
;    0196  1 ! Formal Output Parameters:
;    0197  1 !
;    0198  1 ! None
;    0199  1 !
;    0200  1 ! Side Effects:
;    0201  1 !
;    0202  1 ! None
;    0203  1 !
;    0204  1 ! Completion Codes:
;    0205  1 !
;    0206  1 ! None
;    0207  1 !--
```

```
;    0208  2 BEGIN     ! Routine MEN$Get_Next_State
;    0209  2
;    0210  2  IF (.CRD$GB_Current_State_Table EQL MEN$K_MM_State_Table) THEN
;    0211  3   BEGIN
;    0212  3   CRD$GL_Previous_State = .CRD$GL_MM_Current_State;
;    0213  3      ! Save the old state
;    0214  3   CRD$GL_MM_Current_State =
;    0215  3    .CRD$GAW_MM_State_Table [.CRD$GL_Current_TypeCode, .CRD$GL_MM_Current_State, CRD$K_New_State];
;    0216  3   END
;    0217  3
;    0218  2  ELSE IF (.CRD$GB_Current_State_Table EQL MEN$K_TQ_State_Table) THEN
;    0219  3   BEGIN
;    0220  3   CRD$GL_Previous_State = .CRD$GL_TQ_Current_State;
;    0221  3      ! Save the old state - mainly for debugging purposes
;    0222  3   CRD$GL_TQ_Current_State =
;    0223  3    .CRD$GAW_TQ_State_Table [.CRD$GL_Current_TypeCode, .CRD$GL_TQ_Current_State, CRD$K_New_State];
;    0224  3   END
;    0225  3
;    0226  2  ELSE IF (.CRD$GB_Current_State_Table EQL MEN$K_HS_State_Table) THEN
;    0227  3   BEGIN
;    0228  3   CRD$GL_Previous_State = .CRD$GL_HS_Current_State;
;    0229  3      ! Save the old state - mainly for debugging purposes
;    0230  3   CRD$GL_HS_Current_State =
;    0231  3    .CRD$GAW_HS_State_Table [.CRD$GL_Current_TypeCode, .CRD$GL_HS_Current_State, CRD$K_New_State];
;    0232  3   END
;    0233  3
;    0234  2  ELSE
;    0235  3   $CRD_Print ($ASCIC ('Get_Next_State: Wrong value in current_state_table.!/'))
;    0236  1 END;     ! Routine MEN$Get_Next_State



          .PSECT  DATA,NOWRT,NOEXE,  SHR,2

65  74  61  74  53  5F  74  78  65  4E  5F  74  65  47  35  0000A P.AAB:  .ASCII  \5Get_Next_State: Wrong value in current_\  ;
69  20  65  75  6C  61  76  20  67  6E  6F  72  57  20  3A  00019             ;
      5F  74  6E  65  72  72  75  63  20  6E  00028                          ;
    2F  21  2E  65  6C  62  61  74  5F  65  74  61  74  73  00032    .ASCII  \state_table.!/\         ;

          .EXTRN  CRD$GB_CURRENT_STATE_TABLE
          .EXTRN  CRD$GL_PREVIOUS_STATE
          .EXTRN  CRD$GAW_MM_STATE_TABLE
          .EXTRN  CRD$GL_CURRENT_TYPECODE
          .EXTRN  CRD$GAW_TQ_STATE_TABLE
          .EXTRN  CRD$GAW_HS_STATE_TABLE
          .EXTRN  CRD$PRINT

          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            0000 00000 MEN$GET_NEXT_STATE:
          .WORD   Save nothing          ; 0185
          50 00000000G  EF  9A 00002    MOVZBL  CRD$GB_CURRENT_STATE_TABLE, R0       ; 0210
          42  12 00009    BNEQ    4$                ;
          51 00000000G  EF  D0 0000B    MOVL    CRD$GL_MM_CURRENT_STATE, R1          ; 0212
       00000000G  EF      51  D0 00012    MOVL    R1, CRD$GL_PREVIOUS_STATE          ;
```

```
                50 00000000G  EF 00000050   8F  C5 00019    MULL3   #80, CRD$GL_CURRENT_TYPECODE, R0       ; 0215
                50    6041  DE 00025    MOVAL   (R0)[R1], R0          ;
                26 00000000G  EF  D1 00029    CMPL    CRD$GL_CURRENT_TYPECODE, #38        ;
                05  18 00030    BGEQ    1$              ;
                14       51  D1 00032    CMPL    R1, #20         ;
                05  19 00035    BLSS    2$              ;
                50       01  CE 00037 1$:   MNEGL   #1, R0          ;
                08  11 0003A    BRB     3$              ;
                50 00000000GEF40  9E 0003C 2$:   MOVAB   CRD$GAW_MM_STATE_TABLE[R0], R0       ;
             00000000G  EF 02   A0  3C 00044 3$:   MOVZWL  2(R0), CRD$GL_MM_CURRENT_STATE       ;
              04 0004C    RET                 ; 0208
                01       50  91 0004D 4$:   CMPB    R0, #1          ; 0218
                3E  12 00050    BNEQ    8$              ;
                51 00000000G  EF  D0 00052    MOVL    CRD$GL_TQ_CURRENT_STATE, R1         ; 0220
             00000000G  EF       51  D0 00059    MOVL    R1, CRD$GL_PREVIOUS_STATE       ;
                50 00000000G  EF       28  C5 00060    MULL3   #40, CRD$GL_CURRENT_TYPECODE, R0       ; 0223
                50    6041  DE 00068    MOVAL   (R0)[R1], R0          ;
                26 00000000G  EF  D1 0006C    CMPL    CRD$GL_CURRENT_TYPECODE, #38        ;
                05  18 00073    BGEQ    5$              ;
                0A       51  D1 00075    CMPL    R1, #10         ;
                05  19 00078    BLSS    6$              ;
                50       01  CE 0007A 5$:   MNEGL   #1, R0          ;
                08  11 0007D    BRB     7$              ;
                50 00000000GEF40  9E 0007F 6$:   MOVAB   CRD$GAW_TQ_STATE_TABLE[R0], R0       ;
             00000000G  EF 02   A0  3C 00087 7$:   MOVZWL  2(R0), CRD$GL_TQ_CURRENT_STATE       ;
              04 0008F    RET                 ; 0218
                02       50  91 00090 8$:   CMPB    R0, #2          ; 0226
                42  12 00093    BNEQ    12$             ;
                51 00000000G  EF  D0 00095    MOVL    CRD$GL_HS_CURRENT_STATE, R1         ; 0228
             00000000G  EF       51  D0 0009C    MOVL    R1, CRD$GL_PREVIOUS_STATE       ;
                50 00000000G  EF 00000058   8F  C5 000A3    MULL3   #88, CRD$GL_CURRENT_TYPECODE, R0       ; 0231
                50    6041  DE 000AF    MOVAL   (R0)[R1], R0          ;
                26 00000000G  EF  D1 000B3    CMPL    CRD$GL_CURRENT_TYPECODE, #38        ;
                05  18 000BA    BGEQ    9$              ;
                16       51  D1 000BC    CMPL    R1, #22         ;
                05  19 000BF    BLSS    10$             ;
                50       01  CE 000C1 9$:   MNEGL   #1, R0          ;
                08  11 000C4    BRB     11$             ;
                50 00000000GEF40  9E 000C6 10$:   MOVAB   CRD$GAW_HS_STATE_TABLE[R0], R0       ;
             00000000G  EF 02   A0  3C 000CE 11$:   MOVZWL  2(R0), CRD$GL_HS_CURRENT_STATE       ;
              04 000D6    RET                 ; 0226
             00000000'  EF  9F 000D7 12$:   PUSHAB  P.AAB           ; 0235
                01  DD 000DD    PUSHL   #1              ;
                1A  DD 000DF    PUSHL   #26             ;
             00000000G  FF       03  FB 000E1    CALLS   #3, @CRD$PRINT          ;
              04 000E8    RET                 ; 0236

; Routine Size:  233 bytes,    Routine Base:  CODE + 001E
```

```
;    0237  1 %SBTTL 'MEN$Get_Next_Action_Routine Routine'
;    0238  1
;    0239  1 ROUTINE MEN$Get_Next_Action_Routine : NOVALUE =
;    0240  1
;    0241  1 !++
;    0242  1 ! Functional Description:
;    0243  1 !
;    0244  1 ! Extract the current action routine number from one of the state tables.
;    0245  1 !
;    0246  1 ! Formal Input Parameters:
;    0247  1 !
;    0248  1 ! None
;    0249  1 !
;    0250  1 ! Formal Output Parameters:
;    0251  1 !
;    0252  1 ! None
;    0253  1 !
;    0254  1 ! Side Effects:
;    0255  1 !
;    0256  1 ! None
;    0257  1 !
;    0258  1 ! Completion Codes:
;    0259  1 !
;    0260  1 ! None
;    0261  1 !--
```

```
;    0262  2 BEGIN     ! Routine MEN$Get_Next_Action_Routine
;    0263  2
;    0264  2  IF (.CRD$GB_Current_State_Table EQL MEN$K_MM_State_Table) THEN
;    0265  2   CRD$GL_Current_Action =
;    0266  2    .CRD$GAW_MM_State_Table [.CRD$GL_Current_TypeCode, .CRD$GL_MM_Current_State, CRD$K_Action_Routine]
;    0267  2
;    0268  2  ELSE IF (.CRD$GB_Current_State_Table EQL MEN$K_TQ_State_Table) THEN
;    0269  2   CRD$GL_Current_Action =
;    0270  2    .CRD$GAW_TQ_State_Table [.CRD$GL_Current_TypeCode, .CRD$GL_TQ_Current_State, CRD$K_Action_Routine]
;    0271  2
;    0272  2  ELSE IF (.CRD$GB_Current_State_Table EQL MEN$K_HS_State_Table) THEN
;    0273  2   CRD$GL_Current_Action =
;    0274  2    .CRD$GAW_HS_State_Table [.CRD$GL_Current_TypeCode, .CRD$GL_HS_Current_State, CRD$K_Action_Routine]
;    0275  2
;    0276  2  ELSE
;    0277  3   $CRD_Print ($ASCIC ('Get_Next_Action_Routine: Wrong value in current_state_table.!/'))
;    0278  3
;    0279  1 END;     ! Routine MEN$Get_Next_Action_Routine



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

6F  69  74  63  41  5F  74  78  65  4E  5F  74  65  47  3E  00040 P.AAC:  .ASCII  \>Get_Next_Action_Routine: Wrong value in\  ;
6E  6F  72  57  20  3A  65  6E  69  74  75  6F  52  5F  6E  0004F             ;
      6E  69  20  65  75  6C  61  76  20  67  0005E                 ;
5F  65  74  61  74  73  5F  74  6E  65  72  72  75  63  20  00068    .ASCII  \ current_state_table.!/\        ;
       2F  21  2E  65  6C  62  61  74  00077              ;

            .EXTRN  CRD$GL_CURRENT_ACTION

            .PSECT  CODE,NOWRT,  SHR,  PIC,2

              0000 00000 MEN$GET_NEXT_ACTION_ROUTINE:
            .WORD   Save nothing          ; 0239
            51 00000000G  EF  9A 00002    MOVZBL  CRD$GB_CURRENT_STATE_TABLE, R1       ; 0264
            2F  12 00009    BNEQ    1$            ;
            50 00000000G  EF 00000050   8F  C5 0000B    MULL3   #80, CRD$GL_CURRENT_TYPECODE, R0       ; 0266
            51 00000000G  EF  D0 00017    MOVL    CRD$GL_MM_CURRENT_STATE, R1        ;
            50    6041  DE 0001E    MOVAL   (R0)[R1], R0          ;
            26 00000000G  EF  D1 00022    CMPL    CRD$GL_CURRENT_TYPECODE, #38       ;
            69  18 00029    BGEQ    3$            ;
            14      51  D1 0002B    CMPL    R1, #20           ;
            64  18 0002E    BGEQ    3$            ;
            50 00000000GEF40  9E 00030    MOVAB   CRD$GAW_MM_STATE_TABLE[R0], R0       ;
            67  11 00038    BRB     5$            ;
            01      51  91 0003A 1$:  CMPB    R1, #1            ; 0268
            2B  12 0003D    BNEQ    2$            ;
            50 00000000G  EF      28  C5 0003F    MULL3   #40, CRD$GL_CURRENT_TYPECODE, R0       ; 0270
            51 00000000G  EF  D0 00047    MOVL    CRD$GL_TQ_CURRENT_STATE, R1        ;
            50    6041  DE 0004E    MOVAL   (R0)[R1], R0          ;
            26 00000000G  EF  D1 00052    CMPL    CRD$GL_CURRENT_TYPECODE, #38       ;
            39  18 00059    BGEQ    3$            ;
            0A      51  D1 0005B    CMPL    R1, #10           ;
            34  18 0005E    BGEQ    3$            ;
```

```
          50 00000000GEF40  9E 00060    MOVAB   CRD$GAW_TQ_STATE_TABLE[R0], R0       ;
          37  11 00068    BRB     5$              ;
          02       51  91 0006A 2$:   CMPB    R1, #2          ; 0272
          3A  12 0006D    BNEQ    6$              ;
          50 00000000G  EF 00000058   8F  C5 0006F    MULL3   #88, CRD$GL_CURRENT_TYPECODE, R0       ; 0274
          51 00000000G  EF  D0 0007B    MOVL    CRD$GL_HS_CURRENT_STATE, R1        ;
          50    6041  DE 00082    MOVAL   (R0)[R1], R0            ;
          26 00000000G  EF  D1 00086    CMPL    CRD$GL_CURRENT_TYPECODE, #38       ;
          05  18 0008D    BGEQ    3$              ;
          16       51  D1 0008F    CMPL    R1, #22         ;
          05  19 00092    BLSS    4$              ;
          50       01  CE 00094 3$:   MNEGL   #1, R0          ;
          08  11 00097    BRB     5$              ;
          50 00000000GEF40  9E 00099 4$:   MOVAB   CRD$GAW_HS_STATE_TABLE[R0], R0       ;
       00000000G  EF       60  3C 000A1 5$:   MOVZWL  (R0), CRD$GL_CURRENT_ACTION       ;
        04 000A8    RET                 ; 0273
       00000000'  EF  9F 000A9 6$:   PUSHAB  P.AAC           ; 0277
          01  DD 000AF    PUSHL   #1              ;
          1A  DD 000B1    PUSHL   #26             ;
       00000000G  FF       03  FB 000B3    CALLS   #3, @CRD$PRINT          ;
        04 000BA    RET                 ; 0279

; Routine Size:  187 bytes,    Routine Base:  CODE + 0107
```

```
;    0280  1 %SBTTL 'MEN$ uring_Machine Routine'
;    0281  1
;    0282  1 GLOBAL ROUTINE MEN$Turing_Machine (TypeCode, Length, Address) =
;    0283  1
;    0284  1 !++
;    0285  1 ! Functional Description:
;    0286  1 !
;    0287  1 ! This routine is the main decision-making routine for CRD Menu Test. It simulates a finite-state automata
;    0288  1 ! and uses three state tables to determine tne next 'state" of CRD. These state tables are defined in the
;    0289  1 ! MENSTATE module. There is one state table for use in presenting the Main Menu. The next state table is
;    0290  1 ! used for stepping through the Test Queue Table. The last state table is used for actually running the
;    0291  1 ! diagnostics. Each state table is modeled after the Auto Test state table, with some improvements made
;    0292  1 ! for the sake of clarity. Each state table contains it's own Internal_Error rout'ne, to use when an internal
;    0293  1 ! error occurs while using that table. The initial state for CRD Menu Test is MM$K_State_Start. The longword
;    0294  1 ! at location CRD$GB_Current_State_Table contains the address of one of the three menu state tables. This
;    0295  1 ! address is used whenever a reference is made to one of the three tables. Each odf the tables makes
;    0296  1 ! references to it's own set of action routines. These routines are in the CRDMMACT, CRDTQACT, and the
;    0297  1 ! CRDHSACT modules. Note that MM stands for Main Menu, TQ for Test Queue, and HS for Hardware Support.
;    0298  1 !
;    0299  1 ! This routine is capable of being called recursively. In fac', if an action routine returns a
;    0300  1 ! CRD$K_Repeat_Turing status value, this routine will call itself recursively. This allows multiple actions
;    0301  1 ! to be taken on a single typecode.
;    0302  1 !
;    0303  1 ! Formal Input Parameters:
;    0304  1 !
;    0305  1 ! TypeCode: The DS typecode value.
;    0306  1 ! Length:   The length of the string being printed out by the Resident-DS, or the length of the
;    0307  1 !    buffer in which to put a string (e.g., the length of the CLI data buffer).
;    0308  1 ! Address:  The address of the string being printed out, or the address of where to put the string.
;    0309  1 !
;    0310  1 ! Formal Output Parameters:
;    0311  1 !
;    0312  1 ! None
;    0313  1 !
;    0314  1 ! Side Effects:
;    0315  1 !
;    0316  1 ! Whew!
;    0317  1 !
;    0318  1 ! Completion Codes:
;    0319  1 !
;    0320  1 ! CRD$K_Success: Branch around what was about to be done in the Resident-DS.
;    0321  1 ! CRD$K_Failure: Perform the action that was about to happen in the Resident-DS.
;    0322  1 !--
```

```
;    0323  2 BEGIN     ! Routine MEN$Turing_Machine
;    0324  2  LOCAL
;    0325  2   Do_Nothing,  ! Two temporaries
;    0326  2   Ending_Value;
;    0327  2
;    0328  2  OWN
;    0329  2   Recursion : BYTE INITIAL (0);
;    0330  2
;    0331  2  !+
;    0332  2  ! Call the action routine. Each routine returns a multi-value status number.
;    0333  2  ! One such number is CRD$K_Repeat_Turing. When a routine returns this status
;    0334  2  ! value, it means that this Turing_Machine routine must be called recursively.
;    0335  2  ! This allows multiple actions to be performed on a single typecode (such as
;    0336  2  ! the DS_Prompt typecode). The Load__Com routines return the length of the
;    0337  2  ! loaded DS command in the high word of the returned status value, so check
;    0338  2  ! for the actual status value in the low word. Return the full longword to the
;    0339  2  ! DS_Interface routine. It will return the value to the Resident-DS routine that
;    0340  2  ! called it. So, this length will get passed back to the DS_SuperLine routine
;    0341  2  ! in the SCRIPT module.
;    0342  2  !-
;    0343  2
;    0344  2  MACRO
; M  0345  2   MM_Action_Routine_Case (AR_Name) [] =
; M  0346  2    [%NAME ('MM$K_Action_', AR_Name)]:
; M  0347  2     BEGIN
; M  0348  2     LOCAL
; M  0349  2      Status : VECTOR [2, WORD];
; M  0350  2
; M  0351  2     Status = %NAME ('MM$', AR Name) (%REMAINING);
; M  0352  2
; M  0353  2     IF .Status [0] EQL CRD$K_Repeat_Turing THEN
; M  0354  2      BEGIN
; M  0355  2      Status = MEN$Turing_Machine (.TypeCode, .Length, .Address);
; M  0356  2      Recursion = .Recursion - 1;
; M  0357  2      $DS_Break;
; M  0358  2      RETURN (.Status)
; M  0359  2      END
; M  0360  2     ELSE
; M  0361  2      BEGIN
; M  0362  2      Recursion = .Recursion - 1;
; M  0363  2      $DS_Break;
; M  0364  2      RETURN (.Status)
; M  0365  2      END
; M  0366  2     END
;    0367  2   %;
;    0368  2
;    0369  2  MACRO
; M  0370  2   TQ_Action_Routine_Case (AR_Name) [] =
; M  0371  2    [%NAME ('TQ$K_Action_', AR_Name)]:
; M  0372  2     BEGIN
; M  0373  2     LOCAL
; M  0374  2      Status : VECTOR [2, WORD];
; M  0375  2
; M  0376  2     Status = %NAME ('TQ$', AR_Name) (%REMAINING);
; M  0377  2
```

```
; M 0378  2      IF .Status [0] EQL CRD$K_Repeat_Turing THEN
; M 0379  2       BEGIN
; M 0380  2       Status = MEN$Turing_Machine (.TypeCode, .Length, .Address);
; M 0381  2       Recursion = .Recursion - 1;
; M 0382  2       $DS_Break;
; M 0383  2       RETURN (.Status)
; M 0384  2       END
; M 0385  2      ELSE
; M 0386  2       BEGIN
; M 0387  2       Recursion = .Recursion - 1;
; M 0388  2       $DS_Break;
; M 0389  2       RETURN (.Status)
; M 0390  2       END
; M 0391  2      END
;    0392  2    %;
;    0393  2
;    0394  2   MACRO
; M 0395  2    HS_Action_Routine_Case (AR_Name) [] =
; M 0396  2     [%NAME ('HS$K_Action_', AR_Name)]:
; M 0397  2      BEGIN
; M 0398  2      LOCAL
; M 0399  2       Status : VECTOR [2, WORD];
; M 0400  2
; M 0401  2      Status = %NAME ('HS$', AR_Name) (%REMAINING);
; M 0402  2
; M 0403  2      IF .Status [0] EQL CRD$K_Repeat_Turing THEN
; M 0404  2       BEGIN
; M 0405  2       Status = MEN$Turing_Machine (.TypeCode, .Length, .Address);
; M 0406  2       Recursion = .Recursion - 1;
; M 0407  2       $DS_Break;
; M 0408  2       RETURN (.Status)
; M 0409  2       END
; M 0410  2      ELSE
; M 0411  2       BEGIN
; M 0412  2       Recursion = .Recursion - 1;
; M 0413  2       $DS_Break;
; M 0414  2       RETURN (.Status)
; M 0415  2       END
; M 0416  2      END
;    0417  2    %;
```

```
;   0418  2  $DS_Break;   ! Check for ^C's typed on the terminal
;   0419  2
;   0420  2  IF .CRD$GB_CRD_Debug THEN
; P 0421  2   $CRD_Print (
; P 0422  2       $ASCIC ('!/MEN$Turing_Machine (Recursion=!UL):!/'),
;   0423  2       .Recursion);
;   0424  2
;   0425  2  Recursion = .Recursion + 1;
;   0426  2
;   0427  2  IF .CRD$GB_CRD_Debug THEN
;   0428  3   BEGIN
; P 0429  3   $CRD_Print (
; P 0430  3       $ASCIC ('  - The current typecode, !UL, is '),
;   0431  3       .Typecode);
;   0432  3   CRD$Print_TypeCode_Name (.TypeCode);
; P 0433  3   $CRD_Print
; P 0434  3    (
; P 0435  3    $ASCIC ('  - State Table: !2UL, MM State: !2UL, TQ State: !2UL, ',
; P 0436  3         'HS State: !2UL, Previous: !2UL!/',
; P 0437  3     '  - Action: !2UL, TypeCode: !2UL, Address: !XL(X), Length: !3UL!/',
; P 0438  3     '  - DS_Flags: !XL(X), DSA_Flags: !XL(X)!/'),
; P 0439  3    .CRD$GB_Current_State_Table, .CRD$GL_MM_Current_State, .CRD$GL_TQ_Current_State,
; P 0440  3    .CRD$GL_HS_Current_State, .CRD$GL_Previous_State,
; P 0441  3    .CRD$GL_Current_Action, .TypeCode, .Address, .Length, ..CRD$GA_DS_Flags, .DSA$GL_Flags
;   0442  3    );
;   0443  2   END;
;   0444  2
;   0445  2  CRD$GL_Current_TypeCode = .TypeCode;
;   0446  2
;   0447  2  Ending_Value =
;   0448  3   BEGIN
;   0449  3   IF (.CRD$GB_Current_State_Table EQL MEN$K_MM_State_Table) THEN
;   0450  3    MM$K_Action_Advance_State
;   0451  3   ELSE IF (.CRD$GB_Current_State_Table EQL MEN$K_TQ_State_Table) THEN
;   0452  3    TQ$K_Action_Advance_State
;   0453  3   ELSE
;   0454  3    HS$K_Action_Advance_State
;   0455  2   END;
;   0456  2
;   0457  2  DO
;   0458  3   BEGIN
;   0459  3   !+
;   0460  3   ! These two routines MUST be called in this order, because the Get_Next_State routine alters
;   0461  3   ! a value that the Get_Next_Action routine uses.
;   0462  3   !-
;   0463  3
;   0464  3   MEN$Get_Next_Action_Routine ();
;   0465  3   MEN$Get_Next_State ();
;   0466  3   END
;   0467  2  UNTIL (.CRD$GL_Current_Action NEQ .Ending_Value);
;   0468  2
;   0469  2  IF .CRD$GB_CRD_Debug THEN
;   0470  2   $CRD_Print ($ASCIC ('  - Executing action routine: '));
;   0471  2
;   0472  2  Do_Nothing =
```

```
;    0473  3    BEGIN
;    0474  3    IF (.CRD$GB_Current_State_Table EQL MEN$K_MM_State_Table) THEN
;    0475  3     MM$K_Action_Do_Nothing
;    0476  3    ELSE IF (.CRD$GB_Current_State_Table EQL MEN$K_TQ_State_Table) THEN
;    0477  3     TQ$K_Action_Do_Nothing
;    0478  3    ELSE
;    0479  3     HS$K_Action_Do_Nothing
;    0480  2    END;
;    0481  2
;    0482  2   IF (.CRD$GL_Current_Action EQL .Do_Nothing) THEN
;    0483  3    BEGIN
;    0484  3    Recursion = .Recursion - 1;
;    0485  3    IF .CRD$GB_CRD_Debug THEN
;    0486  3     MEN$Print_Action_Routine (.Do_Nothing);
;    0487  4    RETURN (CRD$K_Success)
;    0488  2    END;
;    0489  2
;    0490  2   IF .CRD$GB_CRD_Debug THEN
;    0491  2    MEN$Print_Action_Routine (.CRD$GL_Current_Action);
;    0492  2
;    0493  2   IF (.CRD$GB_Current_State_Table EQL MEN$K_MM_State_Table) THEN
;    0494  2    CASE .CRD$GL_Current_Action FROM 0 TO (MM$K_Action_Last_Action - 1) OF
;    0495  2     SET
;    0496  2     MM_Action_Routine_Case (Print_Menu_Heading);
;    0497  2     MM_Action_Routine_Case (Load_AutoSizer,      .Length, .Address);
;    0498  2     MM_Action_Routine_Case (Set_Flags_AutoSizer, .Length, .Address);
;    0499  2     MM_Action_Routine_Case (Start_AutoSizer,     .Length, .Address);
;    0500  2     MM_Action_Routine_Case (Build_DDBs);
;    0501  2     MM_Action_Routine_Case (Build_HSTs);
;    0502  2     MM_Action_Routine_Case (Done_Inits);
;    0503  2     MM_Action_Routine_Case (Print_Menu);
;    0504  2     MM_Action_Routine_Case (Change_Table);
;    0505  2     MM_Action_Routine_Case (Menu_Prompt);
;    0506  2     MM_Action_Routine_Case (Internal_Error,      .TypeCode, .Length, .Address);
;    0507  2     MM_Action_Routine_Case (AutoSizer_Error);
;    0508  2
;    0509  2     [MM$K_Action_Do_Nothing
;    0510  2     ,MM$K_Action_Advance_State
;    0511  2     ,MM$K_Action_Dummy_3
;    0512  2     ,MM$K_Action_Dummy_4
;    0513  2     ,MM$K_Action_Dummy_5
;    0514  2     ,MM$K_Action_Dummy_6
;    0515  2     ,MM$K_Action_Dummy_7
;    0516  2     ,MM$K_Action_Dummy_8]:
;    0517  3      BEGIN
;    0518  3      CRD$Internal_Error (CRD$K_Bad_Action_Number, .Length, .Address);
;    0519  3        ! Pass length and address so that we have 3 parameters
;    0520  2      END;
;    0521  2
;    0522  2     [OUTRANGE]:
;    0523  3      BEGIN
;    0524  3      CRD$Internal_Error (CRD$K_Action_Range_Error, .Length, .Address);
;    0525  3       ! Pass length and address so that we have 3 parameters
;    0526  2      END;
;    0527  2     TES
```

```
;    0528  2
;    0529  2  ELSE IF (.CRD$GB_Current_State_Table EQL MEN$K_TQ_State_Table) THEN
;    0530  2
;    0531  2   CASE .CRD$GL_Current_Action FROM 0 TO (TQ$K_Action_Last_Action - 1) OF
;    0532  2    SET
;    0533  2    TQ_Action_Routine_Case (Init_TQ);
;    0534  2    TQ_Action_Routine_Case (Get_DDB);
;    0535  2    TQ_Action_Routine_Case (Change_Table);
;    0536  2    TQ_Action_Routine_Case (Done_Test_Queue);
;    0537  2    TQ_Action_Routine_Case (Internal_Error, .TypeCode, .Length,  Address);
;    0538  2
;    0539  2    [TQ$K_Action_Do_Nothing
;    0540  2    ,TQ$K_Action_Advance_State
;    0541  2    ,TQ$K_Action_Dummy_3
;    0542  2    ,TQ$K_Action_Dummy_4
;    0543  2    ,TQ$K_Action_Dummy_5]:
;    0544  3     BEGIN
;    0545  3     CRD$Internal_Error (CRD$K_Bad_Action_Number, .Length, .Address);
;    0546  3       ! Pass length and address so that we have 3 parameters
;    0547  2     END;
;    0548  2
;    0549  2    [OUTRANGE]:
;    0550  3     BEGIN
;    0551  3     CRD$Internal_Error (CRD$K_Action_Range_Error, .Length, .Address);
;    0552  3      ! Pass length and address so that we have 3 parameters
;    0553  2     END;
;    0554  2    TES
;    0555  2
;    0556  2  ELSE
;    0557  2
;    0558  2   CASE .CRD$GL_Current_Action FROM 0 TO (HS$K_Action_Last_Action - 1) OF
;    0559  2    SET
;    0560  2    HS_Action_Routine_Case (Init_HS);
;    0561  2
;    0562  2    HS_Action_Routine_Case (Advance_Diagnostic);
;    0563  2    HS_Action_Routine_Case (Load_General_Com, .Length, .Address);
;    0564  2    HS_Action_Routine_Case (Advance_General_Com);
;    0565  2    HS_Action_Routine_Case (Done_General_Coms);
;    0566  2
;    0567  2    HS_Action_Routine_Case (Load_Load_Com,  .Length, .Address);
;    0568  2    HS_Action_Routine_Case (Load_Set_Flags_Com, .Length, .Address);
;    0569  2    HS_Action_Routine_Case (Load_Set_Event_Com, .Length, .Address);
;    0570  2    HS_Action_Routine_Case (Load_Select_Com, .Length, .Address);
;    0571  2    HS_Action_Routine_Case (Load_Start_Com,  .Length, .Address);
;    0572  2
;    0573  2    HS_Action_Routine_Case (Change_Table);
;    0574  2    HS_Action_Routine_Case (Load_Error);
;    0575  2    HS_Action_Routine_Case (Start_Error);
;    0576  2    HS_Action_Routine_Case (Diagnostic_Error, .TypeCode, .Length, .Address);
;    0577  2    HS_Action_Routine_Case (Preparation_Error);
;    0578  2    HS_Action_Routine_Case (Internal_Error,  .TypeCode, .Length, .Address);
;    0579  2
;    0580  2    [HS$K_Action_Do_Nothing
;    0581  2    ,HS$K_Action_Advance_State
;    0582  2    ,HS$K_Action_Dummy_3
```

```
;    0583  2     ,HS$K_Action_Dummy_4
;    0584  2     ,HS$K_Action_Dummy_5
;    0585  2     ,HS$K_Action_Dummy_6]:
;    0586  3      BEGIN
;    0587  3      CRD$Internal_Error (CRD$K_Bad_Action_Number, .Length, .Address);
;    0588  3        ! Pass length and address so that we have 3 parameters
;    0589  2      END;
;    0590  2
;    0591  2     [OUTRANGE]:
;    0592  3      BEGIN
;    0593  3      CRD$Internal_Error (CRD$K_Action_Range_Error, .Length, .Address);
;    0594  3       ! Pass length and address so that we have 3 parameters
;    0595  2      END;
;    0596  2     TES;
;    0597  2
;    0598  3   RETURN (CRD$K_Failure)
;    0599  1 END;      ! Routine MEN$Turing_Machine



            .PSECT  WORK,NOEXE,2

        00  00000 RECURSION:
            .BYTE   0           ;

            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

4D  5F  67  6E  69  72  75  54  24  4E  45  4D  2F  21  27  0007F P.AAD:  .ASCII  \'!/MEN$Turing_Machine (Recursion=!UL):!/\  ;
69  73  72  75  63  65  52  28  20  65  6E  69  68  63  61  0008E              ;
      2F  21  3A  29  4C  55  21  3D  6E  6F  0009D                ;
6E  65  72  72  75  63  20  65  68  54  20  2D  20  20  22  000A7 P.AAE:  .ASCII  \'  - The current typecode, !UL, is \       ;
4C  55  21  20  2C  65  64  6F  63  65  70  79  74  20  74  000B6              ;
      20  73  69  20  2C  000C5             ;
6C  62  61  54  20  65  74  61  74  53  20  2D  20  20  C1  000CA P.AAF:  .ASCII  <193>\  - State Table: !2UL, MM State: !\   ;
61  74  53  20  4D  4D  20  2C  4C  55  32  21  20  3A  65  000D9              ;
      21  20  3A  65  74  000E8              ;
20  3A  65  74  61  74  53  20  51  54  20  2C  4C  55  32  000ED    .ASCII  \2UL, TQ State: !2UL, HS State: !2UL, Pre\  ;
3A  65  74  61  74  53  20  53  48  20  2C  4C  55  32  21  000FC              ;
        65  72  50  20  2C  4C  55  32  21  20  0010B                ;
20  20  2F  21  4C  55  32  21  20  3A  73  75  6F  69  76  00115    .ASCII  \vious: !2UL!/  - Action: !2UL, TypeCode:\  ;
2C  4C  55  32  21  20  3A  6E  6F  69  74  63  41  20  2D  00124              ;
      3A  65  64  6F  43  65  70  79  54  20  00133                ;
3A  73  73  65  72  64  64  41  20  2C  4C  55  32  21  20  0013D    .ASCII  \ !2UL, Address: !XL(X), Length: !3UL!/  \  ;
68  74  67  6E  65  4C  20  2C  29  58  28  4C  58  21  20  0014C              ;
      20  20  2F  21  4C  55  33  21  20  3A  0015B                ;
4C  58  21  20  3A  73  67  61  6C  46  5F  53  44  20  2D  00165    .ASCII  \- DS_Flags: !XL(X), DSA_Flags: !XL(X)!/\   ;
3A  73  67  61  6C  46  5F  41  53  44  20  2C  29  58  28  00174              ;
    2F  21  29  58  28  4C  58  21  20  00183                  ;
20  67  6E  69  74  75  63  65  78  45  20  2D  20  20  1E  0018C P.AAG:  .ASCII  <30>\  - Executing action routine: \       ;
3A  65  6E  69  74  75  6F  72  20  6E  6F  69  74  63  61  0019B              ;
        20  001AA               ;

            .EXTRN  DS$BREAK, CRD$GB_CRD_DEBUG
            .EXTRN  CRD$PRINT_TYPECODE_NAME
            .EXTRN  CRD$GA_DS_FLAGS
```

```
        .EXTRN  MM$PRINT_MENU_HEADING
        .EXTRN  MM$LOAD_AUTOSIZER
        .EXTRN  MM$SET_FLAGS_AUTOSIZER
        .EXTRN  MM$START_AUTOSIZER
        .EXTRN  MM$BUILD_DDBS, MM$BUILD_HSTS
        .EXTRN  MM$DONE_INITS, MM$PRINT_MENU
        .EXTRN  MM$CHANGE_TABLE
        .EXTRN  MM$MENU_PROMPT, MM$INTERNAL_ERROR
        .EXTRN  MM$AUTOSIZER_ERROR
        .EXTRN  CRD$INTERNAL_ERROR
        .EXTRN  TQ$INIT_TQ, TQ$GET_DDB
        .EXTRN  TQ$CHANGE_TABLE
        .EXTRN  TQ$DONE_TEST_QUEUE
        .EXTRN  TQ$INTERNAL_ERROR
        .EXTRN  HS$INIT_HS, HS$ADVANCE_DIAGNOSTIC
        .EXTRN  HS$LOAD_GENERAL_COM
        .EXTRN  HS$ADVANCE_GENERAL_COM
        .EXTRN  HS$DONE_GENERAL_COMS
        .EXTRN  HS$LOAD_LOAD_COM
        .EXTRN  HS$LOAD_SET_FLAGS_COM
        .EXTRN  HS$LOAD_SET_EVENT_COM
        .EXTRN  HS$LOAD_SELECT_COM
        .EXTRN  HS$LOAD_START_COM
        .EXTRN  HS$CHANGE_TABLE
        .EXTRN  HS$LOAD_ERROR, HS$START_ERROR
        .EXTRN  HS$DIAGNOSTIC_ERROR
        .EXTRN  HS$PREPARATION_ERROR
        .EXTRN  HS$INTERNAL_ERROR

        .PSECT  CODE,NOWRT,  SHR,  PIC,2

          000C 00000     .ENTRY  MEN$TURING_MACHINE, Save R2,R3        ; 0282
 00000000G  9F       00  FB 00002    CALLS   #0, @#DS$BREAK           ; 0417
     18 00000000G  EF  E9 00009    BLBC    CRD$GB_CRD_DEBUG, 1$        ; 0420
     7E 00000000'  EF  9A 00010    MOVZBL  RECURSION, -(SP)            ; 0423
 00000000'  EF  9F 00017    PUSHAB  P.AAD           ;
     01  DD 0001D    PUSHL   #1          ;
     1A  DD 0001F    PUSHL   #26         ;
 00000000G  FF       04  FB 00021    CALLS   #4, @CRD$PRINT           ;
 00000000'  EF  96 00028 1$:   INCB    RECURSION           ; 0425
     69 00000000G  EF  E9 0002E    BLBC    CRD$GB_CRD_DEBUG, 2$        ; 0427
04   AC  DD 00035    PUSHL   TYPECODE            ; 0431
 00000000'  EF  9F 00038    PUSHAB  P.AAE           ;
     01  DD 0003E    PUSHL   #1          ;
     1A  DD 00040    PUSHL   #26         ;
 00000000G  FF       04  FB 00042    CALLS   #4, @CRD$PRINT           ;
04   AC  DD 00049    PUSHL   TYPECODE            ; 0432
 00000000G  EF       01  FB 0004C    CALLS   #1, CRD$PRINT_TYPECODE_NAME      ;
 0000FE00   9F  DD 00053    PUSHL   @#^X0000FE00            ; 0442
 00000000G  FF  DD 00059    PUSHL   @CRD$GA_DS_FLAGS            ;
08   AC  DD 0005F    PUSHL   LENGTH          ;
0C   AC  DD 00062    PUSHL   ADDRESS         ;
04   AC  DD 00065    PUSHL   TYPECODE            ;
 00000000G  EF  DD 00068    PUSHL   CRD$GL_CURRENT_ACTION       ;
 00000000G  EF  DD 0006E    PUSHL   CRD$GL_PREVIOUS_STATE       ;
```

```
                00000000G  EF  DD 00074    PUSHL   CRD$GL_HS_CURRENT_STATE          ;
                00000000G  EF  DD 0007A    PUSHL   CRD$GL_TQ_CURRENT_STATE          ;
                00000000G  EF  DD 00080    PUSHL   CRD$GL_MM_CURRENT_STATE          ;
                    7E 00000000G  EF  9A 00086    MOVZBL  CRD$GB_CURRENT_STATE_TABLE, -(SP)       ;
                00000000'  EF  9F 0008D    PUSHAB  P.AAF           ;
                    01  DD 00093    PUSHL   #1              ;
                    1A  DD 00095    PUSHL   #26             ;
                00000000G  FF      0E  FB 00097    CALLS   #14, @CRD$PRINT         ;
                    52 04   AC  D0 0009E 2$:   MOVL    TYPECODE, R2            ; 0445
                00000000G  EF      52  D0 000A2    MOVL    R2, CRD$GL_CURRENT_TYPECODE         ;
                    50 00000000G  EF  9A 000A9    MOVZBL  CRD$GB_CURRENT_STATE_TABLE, R0      ; 0449
                    53      01  D0 000B0    MOVL    #1, ENDING_VALUE            ; 0451
               FE8D CF      00  FB 000B3 3$:   CALLS   #0, MEN$GET_NEXT_ACTION_ROUTINE      ; 0464
               FD9F CF      00  FB 000B8    CALLS   #0, MEN$GET_NEXT_STATE          ; 0465
                    53 00000000G  EF  D1 000BD    CMPL    CRD$GL_CURRENT_ACTION, ENDING_VALUE     ; 0467
                    ED  13 000C4    BEQL    3$              ;
                    11 00000000G  EF  E9 000C6    BLBC    CRD$GB_CRD_DEBUG, 4$            ; 0469
                00000000'  EF  9F 000CD    PUSHAB  P.AAG               ; 0470
                    01  DD 000D3    PUSHL   #1              ;
                    1A  DD 000D5    PUSHL   #26             ;
                00000000G  FF      03  FB 000D7    CALLS   #3, @CRD$PRINT          ;
                    50 00000000G  EF  9A 000DE 4$:   MOVZBL  CRD$GB_CURRENT_STATE_TABLE, R0      ; 0474
                    50  D4 000E5    CLRL    DO_NOTHING              ; 0476
                    50 00000000G  EF  D1 000E7    CMPL    CRD$GL_CURRENT_ACTION, DO_NOTHING       ; 0482
                    1A  12 000EE    BNEQ    6$              ;
                00000000'  EF  97 000F0    DECB    RECURSION               ; 0484
                    09 00000000G  EF  E9 000F6    BLBC    CRD$GB_CRD_DEBUG, 5$            ; 0485
                    50  DD 000FD    PUSHL   DO_NOTHING              ; 0486
                00000000V  EF      01  FB 000FF    CALLS   #1, MEN$PRINT_ACTION_ROUTINE        ;
                    50      01  D0 00106 5$:   MOVL    #1, R0              ; 0487
                 04 00109    RET                 ;
                    0D 00000000G  EF  E9 0010A 6$:   BLBC    CRD$GB_CRD_DEBUG, 7$            ; 0490
                00000000G  EF  DD 00111    PUSHL   CRD$GL_CURRENT_ACTION           ; 0491
                00000000V  EF      01  FB 00117    CALLS   #1, MEN$PRINT_ACTION_ROUTINE        ;
                    50 00000000G  EF  D0 0011E 7$:   MOVL    CRD$GL_CURRENT_ACTION, R0       ; 0494
                    51 00000000G  EF  9A 00125    MOVZBL  CRD$GB_CURRENT_STATE_TABLE, R1      ; 0493
                    03  13 0012C    BEQL    8$              ;
                 00BA  31 0012E    BRW     24$             ;
                    13      00      50  CF 00131 8$:   CASEL   R0, #0, #19            ; 0494
           022F         022F      022F     022F     00135 9$:   .WORD   52$-9$,-            ;
           004D         0040      002B     022F     0013D    52$-9$,-           ;
           0070         0067      022F     005A     00145    52$-9$,-           ;
           0082         022F      0079     022F     0014D    52$-9$,-           ;
           00AC         009D      0094     008B     00155    52$-9$,-           ;
                    10$-9$,-            ;
                    13$-9$,             ;
                    14$-9$,             ;
                    15$ 9$,             ;
                    52$ 9$,             ;
                    16$ 9$,             ;
                    17$ 9$,             ;
                    52$ 9$,             ;
                    18$ 9$,             ;
                    52$ 9$,             ;
                    19$ 9$,-            ;
```

```
                20$-9$,-          ;
                21$-9$,-          ;
                22$-9$,-          ;
                23$-9$          ;
        0112  31 0015D     BRW     33$             ; 0524
    00000000G  EF       00  FB 00160 10$:   CALLS   #0, MM$PRINT_MENU_HEADING        ; 0496
        53       50  D0 00167 11$:   MOVL    R0, STATUS          ;
        02       53  B1 0016A     CMPW    STATUS, #2          ;
        03  13 0016D     BEQL    12$             ;
      01E1  31 0016F     BRW     51$             ;
      01D0  31 00172 12$:   BRW     50$             ;
        7E 08    AC  7D 00175 13$:   MOVQ    LENGTH, -(SP)           ; 0497
    00000000G  EF       02  FB 00179     CALLS   #2, MM$LOAD_AUTOSIZER         ;
        E5  11 00180     BRB     11$             ;
        7E 08    AC  7D 00182 14$:   MOVQ    LENGTH, -(SP)           ; 0498
    00000000G  EF       02  FB 00186     CALLS   #2, MM$SET_FLAGS_AUTOSIZER        ;
        D8  11 0018D     BRB     11$             ;
        7E 08    AC  7D 0018F 15$:   MOVQ    LENGTH, -(SP)           ; 0499
    00000000G  EF       02  FB 00193     CALLS   #2, MM$START_AUTOSIZER        ;
        CB  11 0019A     BRB     11$             ;
    00000000G  EF       00  FB 0019C 16$:   CALLS   #0, MM$BUILD_DDBS         ; 0500
        C2  11 001A3     BRB     11$             ;
    00000000G  EF       00  FB 001A5 17$:   CALLS   #0, MM$BUILD_HSTS         ; 0501
        B9  11 001AC     BRB     11$             ;
    00000000G  EF       00  FB 001AE 18$:   CALLS   #0, MM$DONE_INITS         ; 0502
        B0  11 001B5     BRB     11$             ;
    00000000G  EF       00  FB 001B7 19$:   CALLS   #0, MM$PRINT_MENU         ; 0503
        A7  11 001BE     BRB     11$             ;
    00000000G  EF       00  FB 001C0 20$:   CALLS   #0, MM$CHANGE_TABLE          ; 0504
        9E  11 001C7     BRB     11$             ;
    00000000G  EF       00  FB 001C9 21$:   CALLS   #0, MM$MENU_PROMPT         ; 0505
        95  11 001D0     BRB     11$             ;
        7E 08    AC  7D 001D2 22$:   MOVQ    LENGTH, -(SP)           ; 0506
        52  DD 001D6     PUSHL   R2              ;
    00000000G  EF       03  FB 001D8     CALLS   #3, MM$INTERNAL_ERROR         ;
        86  11 001DF     BRB     11$             ;
    00000000G  EF       00  FB 001E1 23$:   CALLS   #0, MM$AUTOSIZER_ERROR        ; 0507
      FF7C  31 001E8     BRW     11$             ;
        01       51  91 001EB 24$:   CMPB    R1, #1          ; 0529
        52  12 001EE     BNEQ    31$             ;
        09       00      50  CF 001F0     CASEL   R0, #0, #9          ; 0531
0016        0170      0170    0170     001F4 25$:   .WORD   52$-25$,-          ;
0034        0170      002A    0020     001FC     52$-25$,-           ;
      003E    0170     00204    52$-25$,-           ;
         26$-25$,-          ;
         27$-25$,-          ;
         28$-25$,-          ;
         52$-25$,-          ;
         29$-25$,-          ;
         52$-25$,-          ;
         30$-25$          ;
        68  11 00208     BRB     33$             ; 0551
    00000000G  EF       00  FB 0020A 26$:   CALLS   #0, TQ$INIT_TQ          ; 0533
      FF53  31 00211     BRW     11$             ;
    00000000G  EF       00  FB 00214 27$:   CALLS   #0, TQ$GET_DDB          ; 0534
```

```
          FF49  31 0021B     BRW     11$                ;
       00000000G  EF       00  FB 0021E 28$:   CALLS   #0, TQ$CHANGE_TABLE          ; 0535
          FF3F  31 00225     BRW     11$                ;
       00000000G  EF       00  FB 00228 29$:   CALLS   #0, TQ$DONE_TEST_QUEUE        ; 0536
          FF35  31 0022F     BRW     11$                ;
            7E 08    AC  7D 00232 30$:   MOVQ    LENGTH, -(SP)           ; 0537
            52  DD 00236     PUSHL   R2                 ;
       00000000G  EF       03  FB 00238    CALLS   #3, TQ$INTERNAL_ERROR          ;
          FF25  31 0023F     BRW     11$                ;
            15          00       50  CF 00242 31$:   CASEL   R0, #0, #21           ; 0558
  0036          011E       011E     011E      00246 32$:   .WORD   52$-32$,-          ;
  0062          0058       004A     0040      0024E    52$-32$,-           ;
  0088          007A       006C     011E      00256    52$-32$,-           ;
  011E          011E       00A4     0096      0025E    34$-32$,-           ;
  00D0          00C6       00BC     00B2      00266    35$-32$,-           ;
          00EA     00E0       0026E    36$-32$,-           ;
             37$-32$,-           ;
             38$-32$,-           ;
             52$-32$,-           ;
             39$-32$,-           ;
             40$-32$,-           ;
             41$-32$,-           ;
             42$-32$,            ;
             43$ 32$,-           ;
             52$ 32$,            ;
             52$ 32$,            ;
             44$ 32$,            ;
             45$ 32$,            ;
             46$-32$,            ;
             47$-32$,-           ;
             48$-32$,-           ;
             49$-32$           ;
       7E 08    AC  7D 00272 33$:   MOVQ    LENGTH, -(SP)           ; 0593
       7E       02  CE 00276    MNEGL   #2, -(SP)          ;
    00EF  31 00279     BRW     53$                ;
00000000G  EF       00  FB 0027C 34$:   CALLS   #0, HS$INIT_HS          ; 0560
    FEE1  31 00283     BRW     11$                ;
00000000G  EF       00  FB 00286 35$:   CALLS   #0, HS$ADVANCE_DIAGNOSTIC        ; 0562
    FED7  31 0028D     BRW     11$                ;
       7E 08    AC  7D 00290 36$:   MOVQ    LENGTH, -(SP)           ; 0563
00000000G  EF       02  FB 00294    CALLS   #2, HS$LOAD_GENERAL_COM        ;
    FEC9  31 0029B     BRW     11$                ;
00000000G  EF       00  FB 0029E 37$:   CALLS   #0, HS$ADVANCE_GENERAL_COM        ; 0564
    FEBF  31 002A5     BRW     11$                ;
00000000G  EF       00  FB 002A8 38$:   CALLS   #0, HS$DONE_GENERAL_COMS        ; 0565
    FEB5  31 002AF     BRW     11$                ;
       7E 08    AC  7D 002B2 39$:   MOVQ    LENGTH, -(SP)           ; 0567
00000000G  EF       02  FB 002B6    CALLS   #2, HS$LOAD_LOAD_COM          ;
    FEA7  31 002BD     BRW     11$                ;
       7E 08    AC  7D 002C0 40$:   MOVQ    LENGTH, -(SP)           ; 0568
00000000G  EF       02  FB 002C4    CALLS   #2, HS$LOAD_SET_FLAGS_COM        ;
    FE99  31 002CB     BRW     11$                ;
       7E 08    AC  7D 002CE 41$:   MOVQ    LENGTH, -(SP)           ; 0569
00000000G  EF       02  FB 002D2    CALLS   #2, HS$LOAD_SET_EVENT_COM        ;
    FE8B  31 002D9     BRW     11$                ;
```

```
                7E 08    AC  7D 002DC 42$:   MOVQ    LENGTH, -(SP)           ; 0570
            00000000G    EF       02  FB 002E0    CALLS   #2, HS$LOAD_SELECT_COM      ;
              FE7D   31 002E7     BRW     11$             ;
                7E 08    AC  7D 002EA 43$:   MOVQ    LENGTH, -(SP)           ; 0571
            00000000G    EF       02  FB 002EE    CALLS   #2, HS$LOAD_START_COM       ;
              FE6F   31 002F5     BRW     11$             ;
            00000000G    EF       00  FB 002F8 44$:   CALLS   #0, HS$CHANGE_TABLE       ; 0573
              FE65   31 002FF     BRW     11$             ;
            00000000G    EF       00  FB 00302 45$:   CALLS   #0, HS$LOAD_ERROR         ; 0574
              FE5B   31 00309     BRW     11$             ;
            00000000G    EF       00  FB 0030C 46$:   CALLS   #0, HS$START_ERROR        ; 0575
              FE51   31 00313     BRW     11$             ;
                7E 08    AC  7D 00316 47$:   MOVQ    LENGTH, -(SP)           ; 0576
                52   DD 0031A     PUSHL   R2              ;
            00000000G    EF       03  FB 0031C    CALLS   #3, HS$DIAGNOSTIC_ERROR     ;
              FE41   31 00323     BRW     11$             ;
            00000000G    EF       00  FB 00326 48$:   CALLS   #0, HS$PREPARATION_ERROR      ; 0577
              FE37   31 0032D     BRW     11$             ;
                7E 08    AC  7D 00330 49$:   MOVQ    LENGTH, -(SP)           ; 0578
                52   DD 00334     PUSHL   R2              ;
            00000000G    EF       03  FB 00336    CALLS   #3, HS$INTERNAL_ERROR       ;
                53       50  D0 0033D     MOVL    R0, STATUS              ;
                02       53  B1 00340     CMPW    STATUS, #2              ;
                0E   12 00343     BNEQ    51$             ;
                7E 08    AC  7D 00345 50$:   MOVQ    LENGTH, -(SP)           ;
                52   DD 00349     PUSHL   R2              ;
        FCB0    CF       03  FB 0034B    CALLS   #3, MEN$TURING_MACHINE      ;
                53       50  D0 00350     MOVL    R0, STATUS              ;
           00000000'    EF  97 00353 51$:   DECB    RECURSION               ;
           00000000G    9F       00  FB 00359    CALLS   #0, @#DS$BREAK          ;
                50       53  D0 00360     MOVL    STATUS, R0              ;
             04 00363     RET                 ;
                7E 08    AC  7D 00364 52$:   MOVQ    LENGTH, -(SP)           ; 0587
                7E       09  CE 00368     MNEGL   #9, -(SP)               ;
            00000000G    EF       03  FB 0036B 53$:   CALLS   #3, CRD$INTERNAL_ERROR      ;
                50   D4 00372     CLRL    R0                  ; 0598
             04 00374     RET                 ; 0599

; Routine Size:  885 bytes,    Routine Base:  CODE + 01C2
```

```
;    0600  1 %SBTTL 'MEN$Print_Action_Routine Routine'
;    0601  1
;    0602  1 GLOBAL ROUTINE MEN$Print_Action_Routine (Action_Routine) : NOVALUE =
;    0603  1
;    0604  1 !++
;    0605  1 ! Functional Description:
;    0606  1 !
;    0607  1 ! This routine prints a message saying what the Action_Routine is.
;    0608  1 !
;    0609  1 ! Formal Input Parameters:
;    0610  1 !
;    0611  1 ! Action_Routine : An action routine constant.
;    0612  1 !
;    0613  1 ! Formal Output Parameters:
;    0614  1 !
;    0615  1 ! None
;    0616  1 !
;    0617  1 ! Side Effects:
;    0618  1 !
;    0619  1 ! None
;    0620  1 !
;    0621  1 ! Completion Codes:
;    0622  1 !
;    0623  1 ! None
;    0624  1 !--
```

```
;   0625  2 BEGIN     ! Routine MEN$Print_Action_Routine
;   0626  2  MACRO
; M 0627  2   MM_Action_Routine_Case (Action_Routine_String) =
; M 0628  2    [%NAME ('MM$K_Action_', Action_Routine_String)]:
; M 0629  2     $CRD_Print ($ASCIC ('MM - (!UL) - ', Action_Routine_String, '!/!/'), .Action_Routine)
;   0630  2   %;
;   0631  2
;   0632  2  MACRO
; M 0633  2   TQ_Action_Routine_Case (Action_Routine_String) =
; M 0634  2    [%NAME ('TQ$K_Action_', Action_Routine_String)]:
; M 0635  2     $CRD_Print ($ASCIC ('TQ - (!UL) - ', Action_Routine_String, '!/!/'), .Action_Routine)
;   0636  2   %;
;   0637  2
;   0638  2  MACRO
; M 0639  2   HS_Action_Routine_Case (Action_Routine_String) =
; M 0640  2    [%NAME ('HS$K_Action_', Action_Routine_String)]:
; M 0641  2     $CRD_Print ($ASCIC ('HS - (!UL) - ', Action_Routine_String, '!/!/'), .Action_Routine)
;   0642  2   %;
```

```
;    0643  2  IF (.CRD$GB_Current_State_Table EQL MEN$K_MM_State_Table) THEN
;    0644  2   CASE .Action_Routine FROM 0 TO (MM$K_Action_Last_Action - 1) OF
;    0645  2    SET
;    0646  2    MM_Action_Routine_Case ('Do_Nothing');
;    0647  2    MM_Action_Routine_Case ('Dummy_3');
;    0648  2    MM_Action_Routine_Case ('Dummy_4');
;    0649  2    MM_Action_Routine_Case ('Dummy_5');
;    0650  2    MM_Action_Routine_Case ('Advance_State');
;    0651  2    MM_Action_Routine_Case ('Print_Menu_Heading');
;    0652  2    MM_Action_Routine_Case ('Load_AutoSizer');
;    0653  2    MM_Action_Routine_Case ('Set_Flags_AutoSizer');
;    0654  2    MM_Action_Routine_Case ('Start_AutoSizer');
;    0655  2    MM_Action_Routine_Case ('Dummy_6');
;    0656  2    MM_Action_Routine_Case ('Build_DDBs');
;    0657  2    MM_Action_Routine_Case ('Build_HSTs');
;    0658  2    MM_Action_Routine_Case ('Dummy_7');
;    0659  2    MM_Action_Routine_Case ('Done_Inits');
;    0660  2    MM_Action_Routine_Case ('Dummy_8');
;    0661  2    MM_Action_Routine_Case ('Print_Menu');
;    0662  2    MM_Action_Routine_Case ('Change_Table');
;    0663  2    MM_Action_Routine_Case ('Menu_Prompt');
;    0664  2    MM_Action_Routine_Case ('Internal_Error');
;    0665  2    MM_Action_Routine_Case ('AutoSizer_Error');
;    0666  2
;    0667  2    [OUTRANGE]:
;    0668  3     BEGIN
;    0669  4     $CRD_Print ($ASCIC ('MM - (!UL) - ???!/'), .Action_Routine)
;    0670  2     END;
;    0671  2    TES
;    0672  2
;    0673  2  ELSE IF (.CRD$GB_Current_State_Table EQL MEN$K_TQ_State_Table) THEN
;    0674  2
;    0675  2   CASE .CRD$GL_Current_Action FROM 0 TO (TQ$K_Action_Last_Action - 1) OF
;    0676  2    SET
;    0677  2    TQ_Action_Routine_Case ('Do_Nothing');
;    0678  2    TQ_Action_Routine_Case ('Advance_State');
;    0679  2    TQ_Action_Routine_Case ('Dummy_3');
;    0680  2    TQ_Action_Routine_Case ('Init_TQ');
;    0681  2    TQ_Action_Routine_Case ('Get_DDB');
;    0682  2    TQ_Action_Routine_Case ('Change_Table');
;    0683  2    TQ_Action_Routine_Case ('Dummy_4');
;    0684  2    TQ_Action_Routine_Case ('Done_Test_Queue');
;    0685  2    TQ_Action_Routine_Case ('Dummy_5');
;    0686  2    TQ_Action_Routine_Case ('Internal_Error');
;    0687  2
;    0688  2    [OUTRANGE]:
;    0689  3     BEGIN
;    0690  4     $CRD_Print ($ASCIC ('TQ - (!UL) - ???!/'), .Action_Routine)
;    0691  2     END;
;    0692  2    TES
;    0693  2
;    0694  2  ELSE IF (.CRD$GB_Current_State_Table EQL MEN$K_HS_State_Table) THEN
;    0695  2
;    0696  2   CASE .CRD$GL_Current_Action FROM 0 TO (HS$K_Action_Last_Action - 1) OF
;    0697  2    SET
```

```
;    0698  2     HS_Action_Routine_Case ('Do_Nothing');
;    0699  2     HS_Action_Routine_Case ('Advance_State');
;    0700  2     HS_Action_Routine_Case ('Dummy_3');
;    0701  2     HS_Action_Routine_Case ('Init_HS');
;    0702  2     HS_Action_Routine_Case ('Advance_Diagnostic');
;    0703  2     HS_Action_Routine_Case ('Load_General_Com');
;    0704  2     HS_Action_Routine_Case ('Advance_General_Com');
;    0705  2     HS_Action_Routine_Case ('Done_General_Coms');
;    0706  2     HS_Action_Routine_Case ('Dummy_4');
;    0707  2     HS_Action_Routine_Case ('Load_Load_Com');
;    0708  2     HS_Action_Routine_Case ('Load_Set_Flags_Com');
;    0709  2     HS_Action_Routine_Case ('Load_Set_Event_Com');
;    0710  2     HS_Action_Routine_Case ('Load_Select_Com');
;    0711  2     HS_Action_Routine_Case ('Load_Start_Com');
;    0712  2     HS_Action_Routine_Case ('Dummy_5');
;    0713  2     HS_Action_Routine_Case ('Dummy_6');
;    0714  2     HS_Action_Routine_Case ('Change_Table');
;    0715  2     HS_Action_Routine_Case ('Load_Error');
;    0716  2     HS_Action_Routine_Case ('Start_Error');
;    0717  2     HS_Action_Routine_Case ('Diagnostic_Error');
;    0718  2     HS_Action_Routine_Case ('Preparation_Error');
;    0719  2     HS_Action_Routine_Case ('Internal_Error');
;    0720  2
;    0721  2     [OUTRANGE]:
;    0722  3      BEGIN
;    0723  4      $CRD_Print ($ASCIC ('HS - (!UL) - ???!/'), .Action_Routine)
;    0724  2      END;
;    0725  2     TES
;    0726  2
;    0727  2   ELSE
; P  0728  2     $CRD_Print ($ASCIC ('?? - (!UL) - State table value wrong, is !UL!/'),
;    0729  2      .Action_Routine, .CRD$GB_Current_State_Table);
;    0730  2
;    0731  1 END;      ! Routine MEN$Print_Action_Routine



            .PSECT  DATA,NOWRT,NOEXE,  SHR,2

44  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1B  001AB P.AAH:  .ASCII  <27>\MM - (!UL) - Do_Nothing!/!/\       ;
 2F  21  2F  21  67  6E  69  68  74  6F  4E  5F  6F  001BA                        ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  18  001C7 P.AAI:  .ASCII  <24>\MM - (!UL) - Dummy_3!/!/\       ;
      2F  21  2F  21  33  5F  79  6D  6D  75  001D6                        ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  18  001E0 P.AAJ:  .ASCII  <24>\MM - (!UL) - Dummy_4!/!/\       ;
      2F  21  2F  21  34  5F  79  6D  6D  75  001EF                        ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  18  001F9 P.AAK:  .ASCII  <24>\MM - (!UL) - Dummy_5!/!/\       ;
      2F  21  2F  21  35  5F  79  6D  6D  75  00208                        ;
41  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1E  00212 P.AAL:  .ASCII  <30>\MM - (!UL) - Advance_State!/!/\       ;
21  2F  21  65  74  61  74  53  5F  65  63  6E  61  76  64  00221                ;
        2F  00230                    ;
50  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  23  00231 P.AAM:  .ASCII  \#MM - (!UL) - Print_Menu_Heading!/!/\      ;
69  64  61  65  48  5F  75  6E  65  4D  5F  74  6E  69  72  00240                ;
          2F  21  2F  21  67  6E  0024F                    ;
4C  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1F  00255 P.AAN:  .ASCII  <31>\MM - (!UL) - Load_AutoSizer!/!/\       ;
2F  21  72  65  7A  69  53  6F  74  75  41  5F  64  61  6F  00264                ;
```

```
                2F  21  00273                   ;
53  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  24  00275 P.AAO:  .ASCII  \$MM - (!UL) - Set_Flags_AutoSizer!/!/\     ;
69  53  6F  74  75  41  5F  73  67  61  6C  46  5F  74  65  00284                 ;
    2F  21  2F  21  72  65  7A  00293                       ;
53  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  20  0029A P.AAP:  .ASCII  \ MM - (!UL) - Start_AutoSizer!/!/\       ;
21  72  65  7A  69  53  6F  74  75  41  5F  74  72  61  74  002A9                 ;
      2F  21  2F  002B8                     ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  18  002BB P.AAQ:  .ASCII  <24>\MM - (!UL) - Dummy_6!/!/\       ;
    2F  21  2F  21  36  5F  79  6D  6D  75  002CA                       ;
42  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1B  002D4 P.AAR:  .ASCII  <27>\MM - (!UL) - Build_DDBs!/!/\       ;
 2F  21  2F  21  73  42  44  44  5F  64  6C  69  75  002E3                   ;
42  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1B  002F0 P.AAS:  .ASCII  <27>\MM - (!UL) - Build_HSTs!/!/\       ;
 2F  21  2F  21  73  54  53  48  5F  64  6C  69  75  002FF                   ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  18  0030C P.AAT:  .ASCII  <24>\MM - (!UL) - Dummy_7!/!/\       ;
    2F  21  2F  21  37  5F  79  6D  6D  75  0031B                       ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1B  00325 P.AAU:  .ASCII  <27>\MM - (!UL) - Done_Inits!/!/\       ;
 2F  21  2F  21  73  74  69  6E  49  5F  65  6E  6F  00334                   ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  18  00341 P.AAV:  .ASCII  <24>\MM - (!UL) - Dummy_8!/!/\       ;
    2F  21  2F  21  38  5F  79  6D  6D  75  00350                       ;
50  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1B  0035A P.AAW:  .ASCII  <27>\MM - (!UL) - Print_Menu!/!/\       ;
 2F  21  2F  21  75  6E  65  4D  5F  74  6E  69  72  00369                   ;
43  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1D  00376 P.AAX:  .ASCII  <29>\MM - (!UL) - Change_Table!/!/\       ;
2F  21  2F  21  65  6C  62  61  54  5F  65  67  6E  61  68  00385                 ;
4D  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1C  00394 P.AAY:  .ASCII  <28>\MM - (!UL) - Menu_Prompt!/!/\       ;
    2F  21  2F  21  74  70  6D  6F  72  50  5F  75  6E  65  003A3                 ;
49  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  1F  003B1 P.AAZ:  .ASCII  <31>\MM - (!UL) - Internal_Error!/!/\       ;
2F  21  72  6F  72  72  45  5F  6C  61  6E  72  65  74  6E  003C0                 ;
            2F  21  003CF                   ;
41  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  20  003D1 P.ABA:  .ASCII  \ MM - (!UL) - AutoSizer_Error!/!/\       ;
21  72  6F  72  72  45  5F  72  65  7A  69  53  6F  74  75  003E0                 ;
      2F  21  2F  003EF                     ;
3F  20  2D  20  29  4C  55  21  28  20  2D  20  4D  4D  12  003F2 P.ABB:  .ASCII  <18>\MM - (!UL) - ???!/\       ;
            2F  21  3F  3F  00401                   ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  1B  00405 P.ABC:  .ASCII  <27>\TQ - (!UL) - Do_Nothing!/!/\       ;
 2F  21  2F  21  67  6E  69  68  74  6F  4E  5F  6F  00414                   ;
41  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  1E  00421 P.ABD:  .ASCII  <30>\TQ - (!UL) - Advance_State!/!/\       ;
21  2F  21  65  74  61  74  53  5F  65  63  6E  61  76  64  00430                 ;
      2F  0043F                     ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  18  00440 P.ABE:  .ASCII  <24>\TQ - (!UL) - Dummy_3!/!/\       ;
    2F  21  2F  21  33  5F  79  6D  6D  75  0044F                       ;
49  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  18  00459 P.ABF:  .ASCII  <24>\TQ - (!UL) - Init_TQ!/!/\       ;
    2F  21  2F  21  51  54  5F  74  69  6E  00468                       ;
47  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  18  00472 P.ABG:  .ASCII  <24>\TQ - (!UL) - Get_DDB!/!/\       ;
    2F  21  2F  21  42  44  44  5F  74  65  00481                       ;
43  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  1D  0048B P.ABH:  .ASCII  <29>\TQ - (!UL) - Change_Table!/!/\       ;
2F  21  2F  21  65  6C  62  61  54  5F  65  67  6E  61  68  0049A                 ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  18  004A9 P.ABI:  .ASCII  <24>\TQ - (!UL) - Dummy_4!/!/\       ;
    2F  21  2F  21  34  5F  79  6D  6D  75  004B8                       ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  20  004C2 P.ABJ:  .ASCII  \ TQ - (!UL) - Done_Test_Queue!/!/\       ;
21  65  75  65  75  51  5F  74  73  65  54  5F  65  6E  6F  004D1                 ;
      2F  21  2F  004E0                     ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  18  004E3 P.ABK:  .ASCII  <24>\TQ - (!UL) - Dummy_5!/!/\       ;
    2F  21  2F  21  35  5F  79  6D  6D  75  004F2                       ;
49  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  1F  004FC P.ABL:  .ASCII  <31>\TQ - (!UL) - Internal_Error!/!/\       ;
2F  21  72  6F  72  72  45  5F  6C  61  6E  72  65  74  6E  0050B                 ;
```

```
          2F  21  0051A                 ;
3F  20  2D  20  29  4C  55  21  28  20  2D  20  51  54  12  0051C P.ABM:  .ASCII  <18>\TQ - (!UL) - ???!/\          ;
          2F  21  3F  3F  0052B                 ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  1B  0052F P.ABN:  .ASCII  <27>\HS - (!UL) - Do_Nothing!/!/\       ;
 2F  21  2F  21  67  6E  69  68  74  6F  4E  5F  6F  0053E                 ;
41  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  1E  0054B P.ABO:  .ASCII  <30>\HS - (!UL) - Advance_State!/!/\       ;
21  2F  21  65  74  61  74  53  5F  65  63  6E  61  76  64  0055A             ;
        2F  00569             ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  18  0056A P.ABP:  .ASCII  <24>\HS - (!UL) - Dummy_3!/!/\        ;
      2F  21  2F  21  33  5F  79  6D  6D  75  00579             ;
49  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  18  00583 P.ABQ:  .ASCII  <24>\HS - (!UL) - Init_HS!/!/\        ;
      2F  21  2F  21  53  48  5F  74  69  6E  00592             ;
41  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  23  0059C P.ABR:  .ASCII  \#HS - (!UL) - Advance_Diagnostic!/!/\      ;
74  73  6F  6E  67  61  69  44  5F  65  63  6E  61  76  64  005AB             ;
          2F  21  2F  21  63  69  005BA                 ;
4C  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  21  005C0 P.ABS:  .ASCII  \!HS - (!UL) - Load_General_Com!/!/\       ;
6D  6F  43  5F  6C  61  72  65  6E  65  47  5F  64  61  6F  005CF             ;
          2F  21  2F  21  005DE                 ;
41  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  24  005E2 P.ABT:  .ASCII  \$HS - (!UL) - Advance_General_Com!/!/\     ;
5F  6C  61  72  65  6E  65  47  5F  65  63  6E  61  76  64  005F1             ;
      2F  21  2F  21  6D  6F  43  00600                 ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  22  00607 P.ABU:  .ASCII  \'HS - (!UL) - Done_General_Coms!/!/\       ;
6D  6F  43  5F  6C  61  72  65  6E  65  47  5F  65  6E  6F  00616             ;
        2F  21  2F  21  73  00625             ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  18  0062A P.ABV:  .ASCII  <24>\HS - (!UL) - Dummy_4!/!/\        ;
      2F  21  2F  21  34  5F  79  6D  6D  75  00639             ;
4C  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  1E  00643 P.ABW:  .ASCII  <30>\HS - (!UL) - Load_Load_Com!/!/\       ;
21  2F  21  6D  6F  43  5F  64  61  6F  4C  5F  64  61  6F  00652             ;
          2F  00661             ;
4C  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  23  00662 P.ABX:  .ASCII  \#HS - (!UL) - Load_Set_Flags_Com!/!/\      ;
43  5F  73  67  61  6C  46  5F  74  65  53  5F  64  61  6F  00671             ;
          2F  21  2F  21  6D  6F  00680                 ;
4C  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  23  00686 P.ABY:  .ASCII  \#HS - (!UL) - Load_Set_Event_Com!/!/\      ;
43  5F  74  6E  65  76  45  5F  74  65  53  5F  64  61  6F  00695             ;
          2F  21  2F  21  6D  6F  006A4                 ;
4C  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  20  006AA P.ABZ:  .ASCII  \ HS - (!UL) - Load_Select_Com!/!/\       ;
21  6D  6F  43  5F  74  63  65  6C  65  53  5F  64  61  6F  006B9             ;
        2F  21  2F  006C8             ;
4C  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  1F  006CB P.ACA:  .ASCII  <31>\HS - (!UL) - Load_Start_Com!/!/\       ;
2F  21  6D  6F  43  5F  74  72  61  74  53  5F  64  61  6F  006DA             ;
            2F  21  006E9             ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  18  006EB P.ACB:  .ASCII  <24>\HS - (!UL) - Dummy_5!/!/\        ;
      2F  21  2F  21  35  5F  79  6D  6D  75  006FA             ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  18  00704 P.ACC:  .ASCII  <24>\HS - (!UL) - Dummy_6!/!/\        ;
      2F  21  2F  21  36  5F  79  6D  6D  75  00713             ;
43  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  1D  0071D P.ACD:  .ASCII  <29>\HS - (!UL) - Change_Table!/!/\       ;
2F  21  2F  21  65  6C  62  61  54  5F  65  67  6E  61  68  0072C             ;
4C  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  1B  0073B P.ACE:  .ASCII  <27>\HS - (!UL) - Load_Error!/!/\       ;
 2F  21  2F  21  72  6F  72  72  45  5F  64  61  6F  0074A                 ;
53  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  1C  00757 P.ACF:  .ASCII  <28>\HS - (!UL) - Start_Error!/!/\       ;
      2F  21  2F  21  72  6F  72  72  45  5F  74  72  61  74  00766             ;
44  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  21  00774 P.ACG:  .ASCII  \!HS - (!UL) - Diagnostic_Error!/!/\       ;
72  6F  72  72  45  5F  63  69  74  73  6F  6E  67  61  69  00783             ;
          2F  21  2F  21  00792                 ;
50  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  22  00796 P.ACH:  .ASCII  \ HS - (!UL) - Preparation_Error!/!/\      ;
```

```
6F  72  72  45  5F  6E  6F  69  74  61  72  61  70  65  72  007A5              ;
     2F  21  2F  21  72  007B4                  ;
49  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  1F  007B9 P.ACI:  .ASCII  <31>\HS - (!UL) - Internal_Error!/!/\       ;
2F  21  72  6F  72  72  45  5F  6C  61  6E  72  65  74  6E  007C8              ;
        2F  21  007D7                  ;
3F  20  2D  20  29  4C  55  21  28  20  2D  20  53  48  12  007D9 P.ACJ:  .ASCII  <18>\HS - (!UL) - ???!/\        ;
        2F  21  3F  3F  007E8                  ;
53  20  2D  20  29  4C  55  21  28  20  2D  20  3F  3F  2E  007EC P.ACK:  .ASCII  \.?? - (!UL) - State table value wrong, i\  ;
75  6C  61  76  20  65  6C  62  61  74  20  65  74  61  74  007FB              ;
      69  20  2C  67  6E  6F  72  77  20  65  0080A                  ;
    2F  21  4C  55  21  20  73  00814     .ASCII  \s !UL!/\          ;



          .PSECT  CODE,NOWRT,  SHR,  PIC,2

            000C 00000     .ENTRY  MEN$PRINT_ACTION_ROUTINE, Save R2,R3       ; 0602
          52 04   AC  D0 00002     MOVL    ACTION_ROUTINE, R2          ; 0644
          53 00000000G  EF  D0 00006    MOVL    CRD$PRINT, R3           ; 0669
          50 00000000G  EF  9A 0000D    MOVZBL  CRD$GB_CURRENT_STATE_TABLE, R0       ; 0643
          03  13 00014     BEQL    1$              ;
        0113  31 00016     BRW     23$             ;
          13        00       52  CF 00019 1$:   CASEL   R2, #0, #19          ; 0644
  0049        003E      005F    0033      0001D 2$:   .WORD   3$-2$,-          ;
  0080        0075      006A    0054      00025     7$-2$,-           ;
  00AC        00A1      0096    008B      0002D     4$-2$,-           ;
  00D8        00CD      00C2    00B7      00035     5$-2$,-           ;
  0104        00F9      00EE    00E3      0003D     6$-2$,-           ;
          8$-2$,-           ;
          9$-2$,-           ;
          10$-2$,-          ;
          11$-2$,-          ;
          12$-2$,-          ;
          13$-2$,-          ;
          14$-2$,-          ;
          15$-2$,-          ;
          16$-2$,-          ;
          17$-2$,-          ;
          18$ 2$,-          ;
          19$ 2$,-          ;
          20$ 2$,-          ;
          21$ 2$,           ;
          22$-2$            ;
       52  DD 00045     PUSHL   R2              ; 0669
00000000'  EF  9F 00047     PUSHAB  P.ABB           ;
  02A3  31 0004D     BRW     61$             ;
       52  DD 00050 3$:   PUSHL   R2              ; 0646
00000000'  EF  9F 00052     PUSHAB  P.AAH           ;
  0298  31 00058     BRW     61$             ;
       52  DD 0005B 4$:   PUSHL   R2              ; 0647
00000000'  EF  9F 0005D     PUSHAB  P.AAI           ;
  028D  31 00063     BRW     61$             ;
       52  DD 00066 5$:   PUSHL   R2              ; 0648
00000000'  EF  9F 00068     PUSHAB  P.AAJ           ;
  0282  31 0006E     BRW     61$             ;
```

```
          52  DD 00071 6$:   PUSHL   R2              ; 0649
    00000000'  EF  9F 00073    PUSHAB  P.AAK               ;
      0277  31 00079    BRW     61$               ;
          52  DD 0007C 7$:   PUSHL   R2              ; 0650
    00000000'  EF  9F 0007E    PUSHAB  P.AAL               ;
      026C  31 00084    BRW     61$               ;
          52  DD 00087 8$:   PUSHL   R2              ; 0651
    00000000'  EF  9F 00089    PUSHAB  P.AAM               ;
      0261  31 0008F    BRW     61$               ;
          52  DD 00092 9$:   PUSHL   R2              ; 0652
    00000000'  EF  9F 00094    PUSHAB  P.AAN               ;
      0256  31 0009A    BRW     61$               ;
          52  DD 0009D 10$:   PUSHL   R2              ; 0653
    00000000'  EF  9F 0009F    PUSHAB  P.AAO               ;
      024B  31 000A5    BRW     61$               ;
          52  DD 000A8 11$:   PUSHL   R2              ; 0654
    00000000'  EF  9F 000AA    PUSHAB  P.AAP               ;
      0240  31 000B0    BRW     61$               ;
          52  DD 000B3 12$:   PUSHL   R2              ; 0655
    00000000'  EF  9F 000B5    PUSHAB  P.AAQ               ;
      0235  31 000BB    BRW     61$               ;
          52  DD 000BE 13$:   PUSHL   R2              ; 0656
    00000000'  EF  9F 000C0    PUSHAB  P.AAR               ;
      022A  31 000C6    BRW     61$               ;
          52  DD 000C9 14$:   PUSHL   R2              ; 0657
    00000000'  EF  9F 000CB    PUSHAB  P.AAS               ;
      021F  31 000D1    BRW     61$               ;
          52  DD 000D4 15$:   PUSHL   R2              ; 0658
    00000000'  EF  9F 000D6    PUSHAB  P.AAT               ;
      0214  31 000DC    BRW     61$               ;
          52  DD 000DF 16$:   PUSHL   R2              ; 0659
    00000000'  EF  9F 000E1    PUSHAB  P.AAU               ;
      0209  31 000E7    BRW     61$               ;
          52  DD 000EA 17$:   PUSHL   R2              ; 0660
    00000000'  EF  9F 000EC    PUSHAB  P.AAV               ;
      01FE  31 000F2    BRW     61$               ;
          52  DD 000F5 18$:   PUSHL   R2              ; 0661
    00000000'  EF  9F 000F7    PUSHAB  P.AAW               ;
      01F3  31 000FD    BRW     61$               ;
          52  DD 00100 19$:   PUSHL   R2              ; 0662
    00000000'  EF  9F 00102    PUSHAB  P.AAX               ;
      01E8  31 00108    BRW     61$               ;
          52  DD 0010B 20$:   PUSHL   R2              ; 0663
    00000000'  EF  9F 0010D    PUSHAB  P.AAY               ;
      01DD  31 00113    BRW     61$               ;
          52  DD 00116 21$:   PUSHL   R2              ; 0664
    00000000'  EF  9F 00118    PUSHAB  P.AAZ               ;
      01D2  31 0011E    BRW     61$               ;
          52  DD 00121 22$:   PUSHL   R2              ; 0665
    00000000'  EF  9F 00123    PUSHAB  P.ABA               ;
      01C7  31 00129    BRW     61$               ;
          01      50  91 0012C 23$:   CMPB    R0, #1          ; 0673
          03  13 0012F    BEQL    24$               ;
      0095  31 00131    BRW     36$               ;
          09        00 00000000G  EF  CF 00134 24$:   CASEL   CRD$GL_CURRENT_ACTION, #0, #9        ; 0675
```

```
  0040        0035      002A    001F     0013C 25$:   .WORD   26$-25$,-          ;
  006C        0061      0056    004B     00144    27$-25$,-          ;
        0082     0077     0014C    28$-25$,-          ;
           29$-25$,-          ;
           30$-25$,-          ;
           31$-25$,-          ;
           32$-25$,-          ;
           33$-25$,-          ;
           34$-25$,-          ;
           35$-25$          ;
        52  DD 00150    PUSHL   R2             ; 0690
  00000000'  EF  9F 00152    PUSHAB  P.ABM            ;
    0198  31 00158    BRW     61$            ;
      52  DD 0015B 26$:   PUSHL   R2             ; 0677
  00000000'  EF  9F 0015D    PUSHAB  P.ABC            ;
    018D  31 00163    BRW     61$            ;
      52  DD 00166 27$:   PUSHL   R2             ; 0678
  00000000'  EF  9F 00168    PUSHAB  P.ABD            ;
    0182  31 0016E    BRW     61$            ;
      52  DD 00171 28$:   PUSHL   R2             ; 0679
  00000000'  EF  9F 00173    PUSHAB  P.ABE            ;
    0177  31 00179    BRW     61$            ;
      52  DD 0017C 29$:   PUSHL   R2             ; 0680
  00000000'  EF  9F 0017E    PUSHAB  P.ABF            ;
    016C  31 00184    BRW     61$            ;
      52  DD 00187 30$:   PUSHL   R2             ; 0681
  00000000'  EF  9F 00189    PUSHAB  P.ABG            ;
    0161  31 0018F    BRW     61$            ;
      52  DD 00192 31$:   PUSHL   R2             ; 0682
  00000000'  EF  9F 00194    PUSHAB  P.ABH            ;
    0156  31 0019A    BRW     61$            ;
      52  DD 0019D 32$:   PUSHL   R2             ; 0683
  00000000'  EF  9F 0019F    PUSHAB  P.ABI            ;
    014B  31 001A5    BRW     61$            ;
      52  DD 001A8 33$:   PUSHL   R2             ; 0684
  00000000'  EF  9F 001AA    PUSHAB  P.ABJ            ;
    0140  31 001B0    BRW     61$            ;
      52  DD 001B3 34$:   PUSHL   R2             ; 0685
  00000000'  EF  9F 001B5    PUSHAB  P.ABK            ;
    0135  31 001BB    BRW     61$            ;
      52  DD 001BE 35$:   PUSHL   R2             ; 0686
  0000G000'  EF  9F 001C0    PUSHAB  P.ABL            ;
    012A  31 001C6    BRW     61$            ;
      02      50  91 001C9 36$:   CMPB    R0, #2          ; 0694
      03  13 001CC    BEQL    37$            ;
    012A  31 001CE    BRW     62$            ;
      15         00 00000000G  EF  CF 001D1 37$:   CASEL   CRD$GL_CURRENT_ACTION, #0, #21       ; 0696
0058        004D      0042    0037     001D9 38$:   .WORD   39$-38$,-          ;
0084        0079      006E    0063     001E1    40$-38$,-          ;
00AE        00A4      009A    008F     001E9    41$-38$,-          ;
00D6        00CC      00C2    00B8     001F1    42$-38$,-          ;
00FE        00F4      00EA    00E0     001F9    43$-38$,          ;
      0112    0108     00201    44$-38$,-          ;
         45$-38$,          ;
         46$-38$,          ;
```

```
            47$-38$,-           ;
            48$-38$,-           ;
            49$-38$,-           ;
            50$-38$,-           ;
            51$-38$,-           ;
            52$-38$,-           ;
            53$-38$,-           ;
            54$-38$,-           ;
            55$-38$,-           ;
            56$-38$,-           ;
            57$-38$,-           ;
            58$-38$,-           ;
            59$-38$,-           ;
            60$-38$             ;
        52  DD 00205     PUSHL   R2              ; 0723
00000000'   EF  9F 00207     PUSHAB  P.ACJ              ;
  00E3  31  0020D     BRW     61$             ;
        52  DD 00210 39$:   PUSHL   R2              ; 0698
00000000'   EF  9F 00212     PUSHAB  P.ABN              ;
  00D8  31  00218     BRW     61$             ;
        52  DD 0021B 40$:   PUSHL   R2              ; 0699
00000000'   EF  9F 0021D     PUSHAB  P.ABO              ;
  00CD  31  00223     BRW     61$             ;
        52  DD 00226 41$:   PUSHL   R2              ; 0700
00000000'   EF  9F 00228     PUSHAB  P.ABP              ;
  00C2  31  0022E     BRW     61$             ;
        52  DD 00231 42$:   PUSHL   R2              ; 0701
00000000'   EF  9F 00233     PUSHAB  P.ABQ              ;
  00B7  31  00239     BRW     61$             ;
        52  DD 0023C 43$:   PUSHL   R2              ; 0702
00000000'   EF  9F 0023E     PUSHAB  P.ABR              ;
  00AC  31  00244     BRW     61$             ;
        52  DD 00247 44$:   PUSHL   R2              ; 0703
00000000'   EF  9F 00249     PUSHAB  P.ABS              ;
  00A1  31  0024F     BRW     61$             ;
        52  DD 00252 45$:   PUSHL   R2              ; 0704
00000000'   EF  9F 00254     PUSHAB  P.ABT              ;
  0096  31  0025A     BRW     61$             ;
        52  DD 0025D 46$:   PUSHL   R2              ; 0705
00000000'   EF  9F 0025F     PUSHAB  P.ABU              ;
  008B  31  00265     BRW     61$             ;
        52  DD 00268 47$:   PUSHL   R2              ; 0706
00000000'   EF  9F 0026A     PUSHAB  P.ABV              ;
  0080  31  00270     BRW     61$             ;
        52  DD 00273 48$:   PUSHL   R2              ; 0707
00000000'   EF  9F 00275     PUSHAB  P.ABW              ;
        76  11  0027B     BRB     61$             ;
        52  DD 0027D 49$:   PUSHL   R2              ; 0708
00000000'   EF  9F 0027F     PUSHAB  P.ABX              ;
        6C  11  00285     BRB     61$             ;
        52  DD 00287 50$:   PUSHL   R2              ; 0709
00000000'   EF  9F 00289     PUSHAB  P.ABY              ;
        62  11  0028F     BRB     61$             ;
        52  DD 00291 51$:   PUSHL   R2              ; 0710
00000000'   EF  9F 00293     PUSHAB  P.ABZ              ;
```

```
            58  11 00299     BRB      61$                ;
            52  DD 0029B 52$:   PUSHL    R2                 ; 0711
      00000000'  EF  9F 0029D     PUSHAB   P.ACA                ;
            4E  11 002A3     BRB      61$                ;
            52  DD 002A5 53$:   PUSHL    R2                 ; 0712
      00000000'  EF  9F 002A7     PUSHAB   P.ACB                ;
            44  11 002AD     BRB      61$                ;
            52  DD 002AF 54$:   PUSHL    R2                 ; 0713
      00000000'  EF  9F 002B1     PUSHAB   P.ACC                ;
            3A  11 002B7     BRB      61$                ;
            52  DD 002B9 55$:   PUSHL    R2                 ; 0714
      00000000'  EF  9F 002BB     PUSHAB   P.ACD                ;
            30  11 002C1     BRB      61$                ;
            52  DD 002C3 56$:   PUSHL    R2                 ; 0715
      00000000'  EF  9F 002C5     PUSHAB   P.ACE                ;
            26  11 002CB     BRB      61$                ;
            52  DD 002CD 57$:   PUSHL    R2                 ; 0716
      00000000'  EF  9F 002CF     PUSHAB   P.ACF                ;
            1C  11 002D5     BRB      61$                ;
            52  DD 002D7 58$:   PUSHL    R2                 ; 0717
      00000000'  EF  9F 002D9     PUSHAB   P.ACG                ;
            12  11 002DF     BRB      61$                ;
            52  DD 002E1 59$:   PUSHL    R2                 ; 0718
      00000000'  EF  9F 002E3     PUSHAB   P.ACH                ;
            08  11 002E9     BRB      61$                ;
            52  DD 002EB 60$:   PUSHL    R2                 ; 0719
      00000000'  EF  9F 002ED     PUSHAB   P.ACI                ;
            01  DD 002F3 61$:   PUSHL    #1                 ;
            1A  DD 002F5     PUSHL    #26                ;
            63      04  FB 002F7     CALLS    #4, (R3)           ;
         04 002FA     RET                 ; 0696
            50  DD 002FB 62$:   PUSHL    R0                 ; 0729
            52  DD 002FD     PUSHL    R2                 ;
      00000000'  EF  9F 002FF     PUSHAB   P.ACK                ;
            01  DD 00305     PUSHL    #1                 ;
            1A  DD 00307     PUSHL    #26                ;
            63      05  FB 00309     CALLS    #5, (R3)           ;
         04 0030C     RET                 ; 0731

; Routine Size:  781 bytes,    Routine Base:  CODE + 0537
```

```
;   0732  1 END     ! End of MENTURING module
;   0733  0 ELUDOM
```

```
;            PSECT SUMMARY
;
; Name    Bytes          Attributes
;
;  DATA          2075  NOVEC,NOWRT,  RD ,NOEXE,  SHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  CODE          2116  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  WORK             1  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
$ASCIC     Unbound             629     635     641
    Macro    Lib02   235     277     422     430     435     470     646     647     648     649
             650     651     652     653     654     655     656     657     658     659
             660     661     662     663     664     665     669     677     678     679
             680     681     682     683     684     685     686     690     698     699
             700     701     702     703     704     705     706     707     708     709
             710     711     712     713     714     715     716     717     718     719
             723     728
$CRD_LITERALS   Macro    Lib03    76
$CRD_PRINT   Unbound             629     635     641
    Macro    Lib03   235     277     421     429     433     470     646     647     648     649
             650     651     652     653     654     655     656     657     658     659
             660     661     662     663     664     665     669     677     678     679
             680     681     682     683     684     685     686     690     698     699
             700     701     702     703     704     705     706     707     708     709
             710     711     712     713     714     715     716     717     718     719
             723     728
$DS_BREAK    Unbound             357     363     382     388     407     413
    Macro    Lib02   418     496     497     498     499     500     501     502     503     504
             505     506     507     533     534     535     536     537     560     562
             563     564     565     567     568     569     570     571     573     574
             575     576     577     578
$DS_DSADEF   Macro    Lib02    75
$DS_TYPEDEF    Macro    Lib02   235     277     423     431     442     470     646     647     648     649
             650     651     652     653     654     655     656     657     658     659
             660     661     662     663     664     665     669     677     678     679
             680     681     682     683     684     685     686     690     698     699
             700     701     702     703     704     705     706     707     708     709
             710     711     712     713     714     715     716     717     718     719
             723     729
$EQULST     Macro    Lib04    76     235     277     423     431     442     470     646     647     648
             649     650     651     652     653     654     655     656     657     658
             659     660     661     662     663     664     665     669     677     678
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ----------------
              679     680     681     682     683     684     685     686     690     698
              699     700     701     702     703     704     705     706     707     708
              709     710     711     712     713     714     715     716     717     718
              719     723     729
$ER    Comptime     92
$MODULE    Bind         92
ACTION_ROUTINE   Unbound          629     635     641
    RoutForm   602   644.    646.    647.    648.    649.    650.    651.    652.    653.    654.
              655.    656.    657.    658.    659.    660.    661.    662.    663.    664.
              665.    669.    677.    678.    679.    680.    681.    682.    683.    684.
              685.    686.    690.    698.    699.    700.    701.    702.    703.    704.
              705.    706.    707.    708.    709.    710.    711.    712.    713.    714.
              715.    716.    717.    718.    719.    723.    729.
ACTION_ROUTINE_STRING  MacrForm   627   628     629
    MacrForm   633   634     635
    MacrForm   639   640     641
ADDRESS    Unbound          355     380     405
    RoutForm   282   442.    496.    497.    498.    499.    500.    501.    502.    503.    504.
              505.    506.    507.    518.    524.    533.    534.    535.    536.    537.
              545.    551.    560.    562.    563.    564.    565.    567.    568.    569.
              570.    571.    573.    574.    575.    576.    577.    578.    587.    593.
AR_NAME    MacrForm   345   346     351
    MacrForm   370   371     376
    MacrForm   395   396     401
AUTO$K_EXIT_AUTOSIZER_ERROR Literal     76
AUTO$K_EXIT_CONTROL_C  Literal     76
AUTO$K_EXIT_DUMMY_2  Literal     76
AUTO$K_EXIT_DUMMY_3  Literal     76
AUTO$K_EXIT_ERRORS_COMPLETE Literal     76
AUTO$K_EXIT_ERRORS_INCOMPLETE Literal     76
AUTO$K_EXIT_INTERNAL_ERROR Literal     76
AUTO$K_EXIT_INV_BASE_SYSTEM Literal     76
AUTO$K_EXIT_LAST_REASON  Literal     76    76
AUTO$K_EXIT_NOERRORS_COMPLETE Literal     76
AUTO$K_EXIT_NOERRORS_INCOMPLETE Literal     76
AUTO$K_EXIT_NOERRORS_PREP Literal     76
CRD$GAW_HS_STATE_TABLE  External Lib03   231     231.    274     274.
CRD$GAW_MM_STATE_TABLE  External Lib03   215     215.    266     266.
CRD$GAW_TQ_STATE_TABLE  External Lib03   223     223.    270     270.
CRD$GA_DS_FLAGS   External Lib03   442a.
CRD$GB_CRD_DEBUG  External Lib03   420.    427.    469.    485.    490.
CRD$GB_CURRENT_STATE_TABLE External Lib03   210.    218.    226.    264.    268.    272.    442.    449.    451.    474.
              476.    493.    529.    643.    673.    694.    729.
CRD$GL_CURRENT_ACTION  External Lib03   265=    269=    273=    442.    467.    482.    491.    494.    531.    558.
              675.    696.
CRD$GL_CURRENT_TYPECODE  External Lib03   215.    223.    231.    266.    270.    274.    445=
CRD$GL_HS_CURRENT_STATE  External Lib03   180=    228.    230=    231.    274.    442.
CRD$GL_MM_CURRENT_STATE  External Lib03   120=    212.    214=    215.    266.    442.
CRD$GL_PREVIOUS_STATE  External Lib03   212=    220=    228=    442.
CRD$GL_TQ_CURRENT_STATE  External Lib03   150=    220.    222=    223.    270.    442.
CRD$INTERNAL_ERROR  ExtRout  Lib03   518c    524c    545c    551c    587c    593c
CRD$K_ACTION_EQL_DO_NOTHING Literal     76
CRD$K_ACTION_RANGE_ERROR Literal     76   524     551     593
```

Symbol	Type	Defined	Referenced ...
CRD$K_ACTION_ROUTINE	Literal	Lib03	266 270 274
CRD$K_ADVANCE_DIAGNOSTIC	Literal		76
CRD$K_ADVANCE_GENERAL_COM	Literal		76
CRD$K_ADVANCE_STATE	Literal		76
CRD$K_ALLOCATION_ERROR	Literal		76
CRD$K_ASSIGN_PTABLE_PTRS	Literal		76
CRD$K_AUTOSIZER_ERROR	Literal		76
CRD$K_AUTOSIZER_SET_FLAGS	Literal		76
CRD$K_BAD_ACTION_NUMBER	Literal		76 518 545 587
CRD$K_BAD_TYPECODE_ERROR	Literal		76
CRD$K_BASE_SYSTEM_IDC	Literal		76
CRD$K_BASE_SYSTEM_LESI	Literal		76
CRD$K_BASE_SYSTEM_NULL	Literal		76
CRD$K_BASE_SYSTEM_UDA_0	Literal		76
CRD$K_BASE_SYSTEM_UDA_1	Literal		76
CRD$K_BASE_SYSTEM_UDA_2	Literal		76
CRD$K_BASE_SYSTEM_UDA_3	Literal		76
CRD$K_CRD_INITIALIZATION	Literal		76
CRD$K_CREATE_HST	Literal		76
CRD$K_DATA_LENGTH_ERROR	Literal		76
CRD$K_DDB_AUTO_SUPPORT_ENTRY	Literal		76
CRD$K_DDB_BLINK	Literal		76
CRD$K_DDB_FLINK	Literal		76
CRD$K_DDB_LAST_ENTRY	Literal		76
CRD$K_DDB_MEMORY_SIZE	Literal		76
CRD$K_DDB_MENU_DEVICE_NUMB	Literal		76
CRD$K_DDB_MENU_SUPPORT_ENTRY	Literal		76
CRD$K_DDB_MENU_SUPPORT_SIZE	Literal		76
CRD$K_DDB_NUMB_SUPPORT_TABLES	Literal		76
CRD$K_DDB_PTABLE_ENTRY	Literal		76
CRD$K_DDB_STATUS	Literal		76
CRD$K_DDB_SUPBLK_BLOCK_PTR	Literal		76
CRD$K_DEVICE_NAME	Literal		76
CRD$K_DIAGNOSTIC_ERROR	Literal		76
CRD$K_DIAGNOSTIC_NAME	Literal		76
CRD$K_DONE_GENERAL_COMS	Literal		76
CRD$K_DO_NOTHING	Literal		76
CRD$K_DUMMY_10	Literal		76
CRD$K_DUMMY_11	Literal		76
CRD$K_DUMMY_12	Literal		76
CRD$K_DUMMY_13	Literal		76
CRD$K_DUMMY_3	Literal		76
CRD$K_DUMMY_4	Literal		76
CRD$K_DUMMY_5	Literal		76
CRD$K_DUMMY_6	Literal		76
CRD$K_DUMMY_7	Literal		76
CRD$K_DUMMY_8	Literal		76
CRD$K_DUMMY_9	Literal		76
CRD$K_EXIT_CRD	Literal		76
CRD$K_FAILURE	Literal	Lib03	598
CRD$K_HOOK_POINT	Literal		76
CRD$K_HS_INTERNAL_ERROR	Literal		76
CRD$K_IDC_EVDLB	Literal		76

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_IDC_EVDLC   Literal     76
CRD$K_IDC_EVDLD   Literal     76
CRD$K_IDC_EVRAD_0  Literal     76
CRD$K_IDC_EVRAD_1  Literal     76
CRD$K_IDC_LAST_DIAGNOSTIC Literal     76
CRD$K_INTERNAL_ERROR  Literal     76
CRD$K_LAST_ACTION_ROUTINE Literal     76
CRD$K_LAST_ENTRY  Literal     76
CRD$K_LAST_FUNCTION  Literal     76
CRD$K_LAST_HOOK_POINT  Literal     76
CRD$K_LAST_INTERNAL_ERROR Literal     76
CRD$K_LAST_TYPE   Literal     76
CRD$K_LESI_ENWCA  Literal     76
CRD$K_LESI_EVRMB_0  Literal     76
CRD$K_LESI_EVRMB_1  Literal     76
CRD$K_LESI_LAST_DIAGNOSTIC Literal     76
CRD$K_LEVEL_4_PASSED  Literal     76
CRD$K_LOAD_AUTOSIZER  Literal     76
CRD$K_LOAD_ERROR  Literal     76
CRD$K_LOAD_GENERAL_COM  Literal     76
CRD$K_LOAD_LOAD_COM  Literal     76
CRD$K_LOAD_SELECT_COM  Literal     76
CRD$K_LOAD_SET_EVENT_COM Literal     76
CRD$K_LOAD_SET_FLAGS_COM Literal     76
CRD$K_LOAD_START_COM  Literal     76
CRD$K_LOAD_SUP_FILE  Literal     76
CRD$K_MM_INTERNAL_ERROR  Literal     76
CRD$K_NEW_LINE   Literal     76
CRD$K_NEW_STATE   Literal  Lib03   215     223     231
CRD$K_PREPARATION_ERROR  Literal     76
CRD$K_PREPARATION_ERROR_TEXT Literal     76
CRD$K_REPEAT_TURING  Unbound          353     378     403
    Literal  Lib03   496     497     498     499     500     501     502     503     504     505
             506     507     533     534     535     536     537     560     562     563
             564     565     567     568     569     570     571     573     574     575
             576     577     578
CRD$K_RUNTIME   Literal     76
CRD$K_SAME_LINE   Literal     76
CRD$K_SET_EVENT_ARGS  Literal     76
CRD$K_SET_FLAGS_ARGS  Literal     76
CRD$K_SET_UP_DIAGNOSTIC  Literal     76
CRD$K_START   Literal     76
CRD$K_START_AUTOSIZER  Literal     76
CRD$K_START_ERROR  Literal     76
CRD$K_START_QUALIFIERS  Literal     76
CRD$K_STAR_LINE   Literal     76
CRD$K_STATE_ADVANCE_DIAGNOSTIC Literal     76
CRD$K_STATE_ADVANCE_GENERAL_COM Literal     76
CRD$K_STATE_ASSIGN_PTABLE_PTRS Literal     76
CRD$K_STATE_AUTOSIZER_ERROR Literal     76
CRD$K_STATE_AUTOSIZER_RUNNING Literal     76
CRD$K_STATE_AUTOSIZER_SET_FLAGS Literal     76
CRD$K_STATE_DIAGNOSTIC_ERROR Literal     76
```

```
Symbol                              Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
CRD$K_STATE_DIAGNOSTIC_RUNNING Literal     76
CRD$K_STATE_DONE_DIAGNOSTICS Literal     76
CRD$K_STATE_DONE_GENERAL_COMS Literal     76
CRD$K_STATE_DUMMY_1  Literal     76
CRD$K_STATE_DUMMY_10  Literal     76
CRD$K_STATE_DUMMY_12  Literal     76
CRD$K_STATE_DUMMY_13  Literal     76
CRD$K_STATE_DUMMY_2  Literal     76
CRD$K_STATE_DUMMY_3  Literal     76
CRD$K_STATE_DUMMY_4  Literal     76
CRD$K_STATE_DUMMY_6  Literal     76
CRD$K_STATE_DUMMY_7  Literal     76
CRD$K_STATE_DUMMY_8  Literal     76
CRD$K_STATE_EXIT_CRD  Literal     76
CRD$K_STATE_INTERNAL_ERROR Literal     76
CRD$K_STATE_LAST_STATE  Literal     76
CRD$K_STATE_LEVEL_4_PASSED Literal     76
CRD$K_STATE_LOAD_AUTOSIZER Literal     76
CRD$K_STATE_LOAD_ERROR  Literal     76
CRD$K_STATE_LOAD_GENERAL_COM Literal     76
CRD$K_STATE_LOAD_LOAD_COM Literal     76
CRD$K_STATE_LOAD_SELECT_COM Literal     76
CRD$K_STATE_LOAD_SET_EVENT_COM Literal     76
CRD$K_STATE_LOAD_SET_FLAGS_COM Literal     76
CRD$K_STATE_LOAD_START_COM Literal     76
CRD$K_STATE_PREPARATION_ERROR Literal     76
CRD$K_STATE_SET_UP_DIAGNOSTIC Literal     76
CRD$K_STATE_START  Literal     76
CRD$K_STATE_START_AUTOSIZER Literal     76
CRD$K_STATE_START_ERROR  Literal     76
CRD$K_SUCCESS   Literal  Lib03   487
CRD$K_TEST_START_TEXT  Literal     76
CRD$K_TQ_INTERNAL_ERROR  Literal     76
CRD$K_TYPECODE   Literal     76
CRD$K_UDA_0_EVDLB  Literal     76
CRD$K_UDA_0_EVDLC  Literal     76
CRD$K_UDA_0_EVDLD  Literal     76
CRD$K_UDA_0_EVRLA_0  Literal     76
CRD$K_UDA_0_LAST_DIAGNOSTIC Literal     76
CRD$K_UDA_1_EVDLB  Literal     76
CRD$K_UDA_1_EVDLC  Literal     76
CRD$K_UDA_1_EVDLD  Literal     76
CRD$K_UDA_1_EVRLA_0  Literal     76
CRD$K_UDA_1_EVRLA_1  Literal     76
CRD$K_UDA_1_LAST_DIAGNOSTIC Literal     76
CRD$K_UDA_2_EVDLB  Literal     76
CRD$K_UDA_2_EVDLC  Literal     76
CRD$K_UDA_2_EVDLD  Literal     76
CRD$K_UDA_2_EVRLA_0  Literal     76
CRD$K_UDA_2_LAST_DIAGNOSTIC Literal     76
CRD$K_UDA_3_EVDLB  Literal     76
CRD$K_UDA_3_EVDLC  Literal     76
CRD$K_UDA_3_EVDLD  Literal     76
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------

CRD$K_UDA_3_EVRLA_0  Literal     76
CRD$K_UDA_3_EVRLA_1  Literal     76
CRD$K_UDA_3_LAST_DIAGNOSTIC Literal     76
CRD$PRINT   External Lib03   235ac   277ac   423ac   431ac   442ac   470ac   646ac   647ac   648ac   649ac
             650ac   651ac   652ac   653ac   654ac   655ac   656ac   657ac   658ac   659ac
             660ac   661ac   662ac   663ac   664ac   665ac   669ac   677ac   678ac   679ac
             680ac   681ac   682ac   683ac   684ac   685ac   686ac   690ac   698ac   699ac
             700ac   701ac   702ac   703ac   704ac   705ac   706ac   707ac   708ac   709ac
             710ac   711ac   712ac   713ac   714ac   715ac   716ac   717ac   718ac   719ac
             723ac   729ac
CRD$PRINT_TYPECODE_NAME  ExtRout  Lib03   432c
DO_NOTHING   Local       325   472=    482.    486.
DS$BREAK   ExtRout    418   418c    496e    496c    497e    497c    498e    498c    499e    499c    500e
             500c    501e    501c    502e    502c    503e    503c    504e    504c    505e
             505c    506e    506c    507e    507c    533e    533c    534e    534c    535e
             535c    536e    536c    537e    537c    560e    560c    562e    562c    563e
             563c    564e    564c    565e    565c    567e    567c    568e    568c    569e
             569c    570e    570c    571e    571c    573e    573c    574e    574c    575e
             575c    576e    576c    577e    577c    578e    578c
DS$K_PRINTB   Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
```

Symbol	Type	Defined	Referenced ...
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		
Literal	716		
Literal	717		
Literal	718		
Literal	719		
Literal	723		
Literal	729		
DS$K_PRINTF	Literal	235	235
Literal	277	277	
Literal	423	423	
Literal	431	431	
Literal	442	442	
Literal	470	470	
Literal	646	646	
Literal	647	647	
Literal	648	648	
Literal	649	649	
Literal	650	650	
Literal	651	651	
Literal	652	652	
Literal	653	653	
Literal	654	654	
Literal	655	655	
Literal	656	656	
Literal	657	657	
Literal	658	658	
Literal	659	659	
Literal	660	660	
Literal	661	661	
Literal	662	662	
Literal	663	663	
Literal	664	664	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     665   665
    Literal     669   669
    Literal     677   677
    Literal     678   678
    Literal     679   679
    Literal     680   680
    Literal     681   681
    Literal     682   682
    Literal     683   683
    Literal     684   684
    Literal     685   685
    Literal     686   686
    Literal     690   690
    Literal     698   698
    Literal     699   699
    Literal     700   700
    Literal     701   701
    Literal     702   702
    Literal     703   703
    Literal     704   704
    Literal     705   705
    Literal     706   706
    Literal     707   707
    Literal     708   708
    Literal     709   709
    Literal     710   710
    Literal     711   711
    Literal     712   712
    Literal     713   713
    Literal     714   714
    Literal     715   715
    Literal     716   716
    L`teral     717   717
    L teral     718   718
    Literal     719   719
    Literal     723   723
    Literal     729   729
DS$K_PRINTI   Literal     235
    Literal      277
    Literal      423
    Literal      431
    Literal      442
    Literal      470
    Literal      646
    Literal      647
    Literal      648
    Literal      649
    Literal      650
    Literal      651
    Literal      652
    Literal      653
    Literal      654
    Literal      655
```

Symbol	Type	Defined	Referenced ...
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		
Literal	716		
Literal	717		
Literal	718		
Literal	719		
Literal	723		
Literal	729		
DS$K_PRINTX	Literal	235	
Literal	277		
Literal	423		
Literal	431		
Literal	442		
Literal	470		
Literal	646		

Symbol	Type	Defined	Referenced ...
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		
Literal	716		
Literal	717		
Literal	718		
Literal	719		

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     723
    Literal     729
DS$K_TYPE_ABORT_PROGRAM  Literal    235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
```

Symbol	Type	Defined	Referenced ...
	Literal		711
	Literal		712
	Literal		713
	Literal		714
	Literal		715
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		723
	Literal		729
DS$K_TYPE_ABORT_TEST	Literal		235
	Literal		277
	Literal		423
	Literal		431
	Literal		442
	Literal		470
	Literal		646
	Literal		647
	Literal		648
	Literal		649
	Literal		650
	Literal		651
	Literal		652
	Literal		653
	Literal		654
	Literal		655
	Literal		656
	Literal		657
	Literal		658
	Literal		659
	Literal		660
	Literal		661
	Literal		662
	Literal		663
	Literal		664
	Literal		665
	Literal		669
	Literal		677
	Literal		678
	Literal		679
	Literal		680
	Literal		681
	Literal		682
	Literal		683
	Literal		684
	Literal		685
	Literal		686
	Literal		690
	Literal		698
	Literal		699
	Literal		700
	Literal		701

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_COMMAND_ERR  Literal    235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
```

Symbol	Type	Defined	Referenced ...
	Literal		683
	Literal		684
	Literal		685
	Literal		686
	Literal		690
	Literal		698
	Literal		699
	Literal		700
	Literal		701
	Literal		702
	Literal		703
	Literal		704
	Literal		705
	Literal		706
	Literal		707
	Literal		708
	Literal		709
	Literal		710
	Literal		711
	Literal		712
	Literal		713
	Literal		714
	Literal		715
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		723
	Literal		729
DS$K_TYPE_COMMAND_OUT	Literal		235
	Literal		277
	Literal		423
	Literal		431
	Literal		442
	Literal		470
	Literal		646
	Literal		647
	Literal		648
	Literal		649
	Literal		650
	Literal		651
	Literal		652
	Literal		653
	Literal		654
	Literal		655
	Literal		656
	Literal		657
	Literal		658
	Literal		659
	Literal		660
	Literal		661
	Literal		662
	Literal		663

Symbol	Type	Defined	Referenced ...
	Literal	664	
	Literal	665	
	Literal	669	
	Literal	677	
	Literal	678	
	Literal	679	
	Literal	680	
	Literal	681	
	Literal	682	
	Literal	683	
	Literal	684	
	Literal	685	
	Literal	686	
	Literal	690	
	Literal	698	
	Literal	699	
	Literal	700	
	Literal	701	
	Literal	702	
	Literal	703	
	Literal	704	
	Literal	705	
	Literal	706	
	Literal	707	
	Literal	708	
	Literal	709	
	Literal	710	
	Literal	711	
	Literal	712	
	Literal	713	
	Literal	714	
	Literal	715	
	Literal	716	
	Literal	717	
	Literal	718	
	Literal	719	
	Literal	723	
	Literal	729	
DS$K_TYPE_CRD_AUTOTEST	Literal	235	235
	Literal	277	277
	Literal	423	423
	Literal	431	431
	Literal	442	442
	Literal	470	470
	Literal	646	646
	Literal	647	647
	Literal	648	648
	Literal	649	649
	Literal	650	650
	Literal	651	651
	Literal	652	652
	Literal	653	653
	Literal	654	654

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     655   655
    Literal     656   656
    Literal     657   657
    Literal     658   658
    Literal     659   659
    Literal     660   660
    Literal     661   661
    Literal     662   662
    Literal     663   663
    Literal     664   664
    Literal     665   665
    Literal     669   669
    Literal     677   677
    Literal     678   678
    Literal     679   679
    Literal     680   680
    Literal     681   681
    Literal     682   682
    Literal     683   683
    Literal     684   684
    Literal     685   685
    Literal     686   686
    Literal     690   690
    Literal     698   698
    Literal     699   699
    Literal     700   700
    Literal     701   701
    Literal     702   702
    Literal     703   703
    Literal     704   704
    Literal     705   705
    Literal     706   706
    Literal     707   707
    Literal     708   708
    Literal     709   709
    Literal     710   710
    Literal     711   711
    Literal     712   712
    Literal     713   713
    Literal     714   714
    Literal     715   715
    Literal     716   716
    Literal     717   717
    Literal     718   718
    Literal     719   719
    Literal     723   723
    Literal     729   729
DS$K_TYPE_DS_PROMPT  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
```

Symbol	Type	Defined	Referenced ...
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		
Literal	716		
Literal	717		
Literal	718		

Symbol	Type	Defined	Referenced ...
	Literal	719	
	Literal	723	
	Literal	729	
DS$K_TYPE_DS_START	Literal	235	
	Literal	277	
	Literal	423	
	Literal	431	
	Literal	442	
	Literal	470	
	Literal	646	
	Literal	647	
	Literal	648	
	Literal	649	
	Literal	650	
	Literal	651	
	Literal	652	
	Literal	653	
	Literal	654	
	Literal	655	
	Literal	656	
	Literal	657	
	Literal	658	
	Literal	659	
	Literal	660	
	Literal	661	
	Literal	662	
	Literal	663	
	Literal	664	
	Literal	665	
	Literal	669	
	Literal	677	
	Literal	678	
	Literal	679	
	Literal	680	
	Literal	681	
	Literal	682	
	Literal	683	
	Literal	684	
	Literal	685	
	Literal	686	
	Literal	690	
	Literal	698	
	Literal	699	
	Literal	700	
	Literal	701	
	Literal	702	
	Literal	703	
	Literal	704	
	Literal	705	
	Literal	706	
	Literal	707	
	Literal	708	
	Literal	709	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_ERRDEV  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
```

Symbol	Type	Defined	Referenced ...
	Literal	701	
	Literal	702	
	Literal	703	
	Literal	704	
	Literal	705	
	Literal	706	
	Literal	707	
	Literal	708	
	Literal	709	
	Literal	710	
	Literal	711	
	Literal	712	
	Literal	713	
	Literal	714	
	Literal	715	
	Literal	716	
	Literal	717	
	Literal	718	
	Literal	719	
	Literal	723	
	Literal	729	
DS$K_TYPE_ERRHARD	Literal	235	
	Literal	277	
	Literal	423	
	Literal	431	
	Literal	442	
	Literal	470	
	Literal	646	
	Literal	647	
	Literal	648	
	Literal	649	
	Literal	650	
	Literal	651	
	Literal	652	
	Literal	653	
	Literal	654	
	Literal	655	
	Literal	656	
	Literal	657	
	Literal	658	
	Literal	659	
	Literal	660	
	Literal	661	
	Literal	662	
	Literal	663	
	Literal	664	
	Literal	665	
	Literal	669	
	Literal	677	
	Literal	678	
	Literal	679	
	Literal	680	
	Literal	681	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_ERROR_BODY  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
```

Symbol	Type	Defined	Referenced ...
	Literal		663
	Literal		664
	Literal		665
	Literal		669
	Literal		677
	Literal		678
	Literal		679
	Literal		680
	Literal		681
	Literal		682
	Literal		683
	Literal		684
	Literal		685
	Literal		686
	Literal		690
	Literal		698
	Literal		699
	Literal		700
	Literal		701
	Literal		702
	Literal		703
	Literal		704
	Literal		705
	Literal		706
	Literal		707
	Literal		708
	Literal		709
	Literal		710
	Literal		711
	Literal		712
	Literal		713
	Literal		714
	Literal		715
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		723
	Literal		729
DS$K_TYPE_ERROR_END	Literal		235
	Literal		277
	Literal		423
	Literal		431
	Literal		442
	Literal		470
	Literal		646
	Literal		647
	Literal		648
	Literal		649
	Literal		650
	Literal		651
	Literal		652
	Literal		653

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_ERRPREP  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
```

Symbol	Type	Defined	Referenced ...
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		
Literal	716		
Literal	717		

Symbol	Type	Defined	Referenced ...
Literal	718		
Literal	719		
Literal	723		
Literal	729		
DS$K_TYPE_ERRSOFT	Literal	235	
Literal	277		
Literal	423		
Literal	431		
Literal	442		
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		

Symbol	Type	Defined	Referenced ...
	Literal		709
	Literal		710
	Literal		711
	Literal		712
	Literal		713
	Literal		714
	Literal		715
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		723
	Literal		729
DS$K_TYPE_ERRSUP	Literal		235
	Literal		277
	Literal		423
	Literal		431
	Literal		442
	Literal		470
	Literal		646
	Literal		647
	Literal		648
	Literal		649
	Literal		650
	Literal		651
	Literal		652
	Literal		653
	Literal		654
	Literal		655
	Literal		656
	Literal		657
	Literal		658
	Literal		659
	Literal		660
	Literal		661
	Literal		662
	Literal		663
	Literal		664
	Literal		665
	Literal		669
	Literal		677
	Literal		678
	Literal		679
	Literal		680
	Literal		681
	Literal		682
	Literal		683
	Literal		684
	Literal		685
	Literal		686
	Literal		690
	Literal		698
	Literal		699

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_ERRSYS  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
```

Symbol	Type	Defined	Referenced ...
	Literal		681
	Literal		682
	Literal		683
	Literal		684
	Literal		685
	Literal		686
	Literal		690
	Literal		698
	Literal		699
	Literal		700
	Literal		701
	Literal		702
	Literal		703
	Literal		704
	Literal		705
	Literal		706
	Literal		707
	Literal		708
	Literal		709
	Literal		710
	Literal		711
	Literal		712
	Literal		713
	Literal		714
	Literal		715
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		723
	Literal		729
DS$K_TYPE_ERR_HALT	Literal	235	
	Literal		277
	Literal		423
	Literal		431
	Literal		442
	Literal		470
	Literal		646
	Literal		647
	Literal		648
	Literal		649
	Literal		650
	Literal		651
	Literal		652
	Literal		653
	Literal		654
	Literal		655
	Literal		656
	Literal		657
	Literal		658
	Literal		659
	Literal		660
	Literal		661

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     693
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_EXCEPTION  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_EXCEPTION_HEAD Literal    235
    Literal     277
    Literal     423
    Literal     431
```

Symbol	Type	Defined	Referenced ...
Literal	442		
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		
Literal	716		

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_FIRST_PASS  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_GENERAL  Literal    235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_GENERAL_ERROR  Literal    235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
```

Symbol	Type	Defined	Referenced ...
	Literal	680	
	Literal	681	
	Literal	682	
	Literal	683	
	Literal	684	
	Literal	685	
	Literal	686	
	Literal	690	
	Literal	698	
	Literal	699	
	Literal	700	
	Literal	701	
	Literal	702	
	Literal	703	
	Literal	704	
	Literal	705	
	Literal	706	
	Literal	707	
	Literal	708	
	Literal	709	
	Literal	710	
	Literal	711	
	Literal	712	
	Literal	713	
	Literal	714	
	Literal	715	
	Literal	716	
	Literal	717	
	Literal	718	
	Literal	719	
	Literal	723	
	Literal	729	
DS$K_TYPE_NO_TESTS	Literal	235	
	Literal	277	
	Literal	423	
	Literal	431	
	Literal	442	
	Literal	470	
	Literal	646	
	Literal	647	
	Literal	648	
	Literal	649	
	Literal	650	
	Literal	651	
	Literal	652	
	Literal	653	
	Literal	654	
	Literal	655	
	Literal	656	
	Literal	657	
	Literal	658	
	Literal	659	
	Literal	660	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_PARAM_ERROR  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
```

Symbol	Type	Defined	Referenced ...
	Literal		652
	Literal		653
	Literal		654
	Literal		655
	Literal		656
	Literal		657
	Literal		658
	Literal		659
	Literal		660
	Literal		661
	Literal		662
	Literal		663
	Literal		664
	Literal		665
	Literal		669
	Literal		677
	Literal		678
	Literal		679
	Literal		680
	Literal		681
	Literal		682
	Literal		683
	Literal		684
	Literal		685
	Literal		686
	Literal		690
	Literal		698
	Literal		699
	Literal		700
	Literal		701
	Literal		702
	Literal		703
	Literal		704
	Literal		705
	Literal		706
	Literal		707
	Literal		708
	Literal		709
	Literal		710
	Literal		711
	Literal		712
	Literal		713
	Literal		714
	Literal		715
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		723
	Literal		729
DS$K_TYPE_PROGRAM_END	Literal		235
	Literal		277
	Literal		423

Symbol	Type	Defined	Referenced ...
Literal	431		
Literal	442		
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		

Symbol	Type	Defined	Referenced ...
Literal	716		
Literal	717		
Literal	718		
Literal	719		
Literal	723		
Literal	729		
DS$K_TYPE_PROGRAM_INFO	Literal	235	
Literal	277		
Literal	423		
Literal	431		
Literal	442		
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		

Symbol	Type	Defined	Referenced ...
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		
Literal	716		
Literal	717		
Literal	718		
Literal	719		
Literal	723		
Literal	729		
DS$K_TYPE_PROGRAM_START	Literal	235	
Literal	277		
Literal	423		
Literal	431		
Literal	442		
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		

Symbol	Type	Defined	Referenced ...
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		
Literal	716		
Literal	717		
Literal	718		
Literal	719		
Literal	723		
Literal	729		
DS$K_TYPE_QIO_INVADP	Literal	235	
Literal	277		
Literal	423		
Literal	431		
Literal	442		
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		

Symbol	Type	Defined	Referenced ...
	Literal		679
	Literal		680
	Literal		681
	Literal		682
	Literal		683
	Literal		684
	Literal		685
	Literal		686
	Literal		690
	Literal		698
	Literal		699
	Literal		700
	Literal		701
	Literal		702
	Literal		703
	Literal		704
	Literal		705
	Literal		706
	Literal		707
	Literal		708
	Literal		709
	Literal		710
	Literal		711
	Literal		712
	Literal		713
	Literal		714
	Literal		715
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		723
	Literal		729
DS$K_TYPE_QIO_NODRIVER	Literal		235
	Literal		277
	Literal		423
	Literal		431
	Literal		442
	Literal		470
	Literal		646
	Literal		647
	Literal		648
	Literal		649
	Literal		650
	Literal		651
	Literal		652
	Literal		653
	Literal		654
	Literal		655
	Literal		656
	Literal		657
	Literal		658
	Literal		659

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_QIO_WRONGVER  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
```

Symbol	Type	Defined	Referenced ...
	Literal	651	
	Literal	652	
	Literal	653	
	Literal	654	
	Literal	655	
	Literal	656	
	Literal	657	
	Literal	658	
	Literal	659	
	Literal	660	
	Literal	661	
	Literal	662	
	Literal	663	
	Literal	664	
	Literal	665	
	Literal	669	
	Literal	677	
	Literal	678	
	Literal	679	
	Literal	680	
	Literal	681	
	Literal	682	
	Literal	683	
	Literal	684	
	Literal	685	
	Literal	686	
	Literal	690	
	Literal	698	
	Literal	699	
	Literal	700	
	Literal	701	
	Literal	702	
	Literal	703	
	Literal	704	
	Literal	705	
	Literal	706	
	Literal	707	
	Literal	708	
	Literal	709	
	Literal	710	
	Literal	711	
	Literal	712	
	Literal	713	
	Literal	714	
	Literal	715	
	Literal	716	
	Literal	717	
	Literal	718	
	Literal	719	
	Literal	723	
	Literal	729	
DS$K_TYPE_SCRIPT_ECHO	Literal	235	
	Literal	277	

Symbol	Type	Defined	Referenced ...
Literal	423		
Literal	431		
Literal	442		
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		

Symbol	Type	Defined	Referenced ...
Literal	715		
Literal	716		
Literal	717		
Literal	718		
Literal	719		
Literal	723		
Literal	729		
DS$K_TYPE_SCRIPT_PNF Literal	235		
Literal	277		
Literal	423		
Literal	431		
Literal	442		
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		

Symbol	Type	Defined	Referenced ...
	Literal		706
	Literal		707
	Literal		708
	Literal		709
	Literal		710
	Literal		711
	Literal		712
	Literal		713
	Literal		714
	Literal		715
	Literal		716
	Literal		717
	Literal		718
	Literal		719
	Literal		723
	Literal		729
DS$K_TYPE_SCRIPT_PROMPT	Literal		235
	Literal		277
	Literal		423
	Literal		431
	Literal		442
	Literal		470
	Literal		646
	Literal		647
	Literal		648
	Literal		649
	Literal		650
	Literal		651
	Literal		652
	Literal		653
	Literal		654
	Literal		655
	Literal		656
	Literal		657
	Literal		658
	Literal		659
	Litera		660
	Literal		661
	Literal		662
	Literal		663
	Literal		664
	Literal		665
	Literal		669
	Literal		677
	Literal		678
	Literal		679
	Literal		680
	Literal		681
	Literal		682
	Literal		683
	Literal		684
	Literal		685
	Literal		686

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_SCRIPT_SKIP  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_SEQUENCE_ERROR Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
```

Symbol	Type	Defined	Referenced ...
	Literal	659	
	Literal	660	
	Literal	661	
	Literal	662	
	Literal	663	
	Literal	664	
	Literal	665	
	Literal	669	
	Literal	677	
	Literal	678	
	Literal	679	
	Literal	680	
	Literal	681	
	Literal	682	
	Literal	683	
	Literal	684	
	Literal	685	
	Literal	686	
	Literal	690	
	Literal	698	
	Literal	699	
	Literal	700	
	Literal	701	
	Literal	702	
	Literal	703	
	Literal	704	
	Literal	705	
	Literal	706	
	Literal	707	
	Literal	708	
	Literal	709	
	Literal	710	
	Literal	711	
	Literal	712	
	Literal	713	
	Literal	714	
	Literal	715	
	Literal	716	
	Literal	717	
	Literal	718	
	Literal	719	
	Literal	723	
	Literal	729	
DS$K_TYPE_START_ERR	Literal	235	
	Literal	277	
	Literal	423	
	Literal	431	
	Literal	442	
	Literal	470	
	Literal	646	
	Literal	647	
	Literal	648	
	Literal	649	

Symbol	Type	Defined	Referenced ...
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		
Literal	714		
Literal	715		
Literal	716		
Literal	717		
Literal	718		
Literal	719		
Literal	723		
Literal	729		
DS$K_TYPE_START_LIST	Literal	235	

Symbol	Type	Defined	Referenced ...
Literal	277		
Literal	423		
Literal	431		
Literal	442		
Literal	470		
Literal	646		
Literal	647		
Literal	648		
Literal	649		
Literal	650		
Literal	651		
Literal	652		
Literal	653		
Literal	654		
Literal	655		
Literal	656		
Literal	657		
Literal	658		
Literal	659		
Literal	660		
Literal	661		
Literal	662		
Literal	663		
Literal	664		
Literal	665		
Literal	669		
Literal	677		
Literal	678		
Literal	679		
Literal	680		
Literal	681		
Literal	682		
Literal	683		
Literal	684		
Literal	685		
Literal	686		
Literal	690		
Literal	698		
Literal	699		
Literal	700		
Literal	701		
Literal	702		
Literal	703		
Literal	704		
Literal	705		
Literal	706		
Literal	707		
Literal	708		
Literal	709		
Literal	710		
Literal	711		
Literal	712		
Literal	713		

Symbol	Type	Defined	Referenced ...
	Literal	714	
	Literal	715	
	Literal	716	
	Literal	717	
	Literal	718	
	Literal	719	
	Literal	723	
	Literal	729	
DS$K_TYPE_SUMMARY	Literal	235	
	Literal	277	
	Literal	423	
	Literal	431	
	Literal	442	
	Literal	470	
	Literal	646	
	Literal	647	
	Literal	648	
	Literal	649	
	Literal	650	
	Literal	651	
	Literal	652	
	Literal	653	
	Literal	654	
	Literal	655	
	Literal	656	
	Literal	657	
	Literal	658	
	Literal	659	
	Literal	660	
	Literal	661	
	Literal	662	
	Literal	663	
	Literal	664	
	Literal	665	
	Literal	669	
	Literal	677	
	Literal	678	
	Literal	679	
	Literal	680	
	Literal	681	
	Literal	682	
	Literal	683	
	Literal	684	
	Literal	685	
	Literal	686	
	Literal	690	
	Literal	698	
	Literal	699	
	Literal	700	
	Literal	701	
	Literal	702	
	Literal	703	
	Literal	704	

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DS$K_TYPE_USER_PROMPT  Literal     235
    Literal     277
    Literal     423
    Literal     431
    Literal     442
    Literal     470
    Literal     646
    Literal     647
    Literal     648
    Literal     649
    Literal     650
    Literal     651
    Literal     652
    Literal     653
    Literal     654
    Literal     655
    Literal     656
    Literal     657
    Literal     658
    Literal     659
    Literal     660
    Literal     661
    Literal     662
    Literal     663
    Literal     664
    Literal     665
    Literal     669
    Literal     677
    Literal     678
    Literal     679
    Literal     680
    Literal     681
    Literal     682
    Literal     683
    Literal     684
    Literal     685
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
    Literal     686
    Literal     690
    Literal     698
    Literal     699
    Literal     700
    Literal     701
    Literal     702
    Literal     703
    Literal     704
    Literal     705
    Literal     706
    Literal     707
    Literal     708
    Literal     709
    Literal     710
    Literal     711
    Literal     712
    Literal     713
    Literal     714
    Literal     715
    Literal     716
    Literal     717
    Literal     718
    Literal     719
    Literal     723
    Literal     729
DSA$AL_APTMAIL   Bind         75     75
DSA$GL_APTCOM    Bind         75
DSA$GL_DEVLEN    Bind         75
DSA$GL_DEVNAM    Bind         75
DSA$GL_ERRNO     Bind         75
DSA$GL_EVENT     Bind         75
DSA$GL_FLAGS     Bind         75   442
DSA$GL_MSGPTR    Bind         75
DSA$GL_MSGTYP    Bind         75
DSA$GL_PASSES    Bind         75
DSA$GL_PASSNO    Bind         75
DSA$GL_SECTNO    Bind         75
DSA$GL_SID    Bind          75
DSA$GL_SUBTNO    Bind         75
DSA$GL_TESTNO    Bind         75
DSA$GL_UNITS     Bind         75
ENDING_VALUE     Local       326   447=    467.
GET1ST_      Macro    Lib04     76      235      277      423      431      442      470      646      647      648
                649      650      651      652      653      654      655      656      657      658
                659      660      661      662      663      664      665      669      677      678
                679      680      681      682      683      684      685      686      690      698
                699      700      701      702      703      704      705      706      707      708
                709      710      711      712      713      714      715      716      717      718
                719      723      729
GET2ND_      Unbound             76      235      277      423      431      442      470      646      647      648
                649      650      651      652      653      654      655      656      657      658
                659      660      661      662      663      664      665      669      677      678
```

Symbol Type Defined Referenced ...

679 680 681 682 683 684 685 686 690 698
699 700 701 702 703 704 705 706 707 708
709 710 711 712 713 714 715 716 717 718
719 723 729

HS$ADVANCE_DIAGNOSTIC ExtRout Lib03 562c
HS$ADVANCE_GENERAL_COM ExtRout Lib03 564c
HS$CHANGE_TABLE ExtRout Lib03 573c
HS$DIAGNOSTIC_ERROR ExtRout Lib03 576c
HS$DONE_GENERAL_COMS ExtRout Lib03 565c
HS$INIT_HS ExtRout Lib03 560c
HS$INTERNAL_ERROR ExtRout Lib03 578c
HS$K_ACTION_ADVANCE_DIAGNOSTIC Literal 76 562 702
HS$K_ACTION_ADVANCE_GENERAL_COM Literal 76 564 704
HS$K_ACTION_ADVANCE_STATE Literal 76 454 581 699
HS$K_ACTION_CHANGE_TABLE Literal 76 573 714
HS$K_ACTION_DIAGNOSTIC_ERROR Literal 76 576 717
HS$K_ACTION_DONE_GENERAL_COMS Literal 76 565 705
HS$K_ACTION_DO_NOTHING Literal 76 479 580 698
HS$K_ACTION_DUMMY_3 Literal 76 582 700
HS$K_ACTION_DUMMY_4 Literal 76 583 706
HS$K_ACTION_DUMMY_5 Literal 76 584 712
HS$K_ACTION_DUMMY_6 Literal 76 585 713
HS$K_ACTION_INIT_HS Literal 76 560 701
HS$K_ACTION_INTERNAL_ERROR Literal 76 578 719
HS$K_ACTION_LAST_ACTION Literal 76 558 696
HS$K_ACTION_LOAD_ERROR Literal 76 574 715
HS$K_ACTION_LOAD_GENERAL_COM Literal 76 563 703
HS$K_ACTION_LOAD_LOAD_COM Literal 76 567 707
HS$K_ACTION_LOAD_SELECT_COM Literal 76 570 710
HS$K_ACTION_LOAD_SET_EVENT_COM Literal 76 569 709
HS$K_ACTION_LOAD_SET_FLAGS_COM Literal 76 568 708
HS$K_ACTION_LOAD_START_COM Literal 76 571 711
HS$K_ACTION_PREPARATION_ERROR Literal 76 577 718
HS$K_ACTION_START_ERROR Literal 76 575 716
HS$K_STATE_ADVANCE_DIAGNOSTIC Literal 76
HS$K_STATE_ADVANCE_GENERAL_COM Literal 76
HS$K_STATE_CHANGE_TABLE Literal 76
HS$K_STATE_DIAGNOSTIC_ERROR Literal 76
HS$K_STATE_DIAGNOSTIC_RUNNING Literal 76
HS$K_STATE_DONE_GENERAL_COMS Literal 76
HS$K_STATE_DUMMY_1 Literal 76
HS$K_STATE_DUMMY_2 Literal 76
HS$K_STATE_DUMMY_3 Literal 76
HS$K_STATE_DUMMY_4 Literal 76
HS$K_STATE_DUMMY_6 Literal 76
HS$K_STATE_INIT_HS Literal 76 180
HS$K_STATE_INTERNAL_ERROR Literal 76
HS$K_STATE_LAST_STATE Literal 76
HS$K_STATE_LOAD_ERROR Literal 76
HS$K_STATE_LOAD_GENERAL_COM Literal 76
HS$K_STATE_LOAD_LOAD_COM Literal 76
HS$K_STATE_LOAD_SELECT_COM Literal 76
HS$K_STATE_LOAD_SET_EVENT_COM Literal 76

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
HS$K_STATE_LOAD_SET_FLAGS_COM Literal     76
HS$K_STATE_LOAD_START_COM Literal     76
HS$K_STATE_PREPARATION_ERROR Literal     76
HS$K_STATE_START_ERROR  Literal     76
HS$LOAD_ERROR   ExtRout  Lib03   574c
HS$LOAD_GENERAL_COM  ExtRout  Lib03   563c
HS$LOAD_LOAD_COM  ExtRout  Lib03   567c
HS$LOAD_SELECT_COM  ExtRout  Lib03   570c
HS$LOAD_SET_EVENT_COM  ExtRout  Lib03   569c
HS$LOAD_SET_FLAGS_COM  ExtRout  Lib03   568c
HS$LOAD_START_COM  ExtRout  Lib03   571c
HS$PREPARATION_ERROR  ExtRout  Lib03   577c
HS$START_ERROR   ExtRout  Lib03   575c
HS_ACTION_ROUTINE_CASE  Macro      395   560     562     563     564     565     567     568     569     570     571
             573     574     575     576     577     578
    Macro     639   698     699     700     701     702     703     704     705     706     707
             708     709     710     711     712     713     714     715     716     717
             718     719
LENGTH    Unbound          355     380     405
    RoutForm   282   442.    496.    497.    498.    499.    500.    501.    502.    503.    504.
             505.    506.    507.    518.    524.    533.    534.    535.    536.    537.
             545.    551.    560.    562.    563.    564.    565.    567.    568.    569.
             570.    571.    573.    574.    575.    576.    577.    578.    587.    593.
MEN$GET_NEXT_ACTION_ROUTINE Routine    239    67f    464c
MEN$GET_NEXT_STATE  Routine    185    65f    465c
MEN$INIT_HS_STATE_TABLE  GlobRout   155 Lib03e     63f
MEN$INIT_MM_STATE_TABLE  GlobRout    95 Lib03e     61f
MEN$INIT_TQ_STATE_TABLE  GlobRout   125 Lib03e     62f
MEN$K_HS_STATE_TABLE  Literal     76   226     272     694
MEN$K_MM_STATE_TABLE  Literal     76   210     264     449     474     493     643
MEN$K_TQ_STATE_TABLE  Literal     76   218     268     451     476     529     673
MEN$PRINT_ACTION_ROUTINE GlobRout   602 Lib03e     71f    486c    491c
MEN$TURING_MACHINE  Unbound          355     380     405
    GlobRout   282 Lib03e     69f    496c    497c    498c    499c    500c    501c    502c    503c
             504c    505c    506c    507c    533c    534c    535c    536c    537c    560c
             562c    563c    564c    565c    567c    568c    569c    570c    571c    573c
             574c    575c    576c    577c    578c
MENU$K_EXIT_AUTOSIZER_ERROR Literal     76
MENU$K_EXIT_INTERNAL_ERROR Literal     76
MENU$K_EXIT_LAST_REASON  Literal     76
MENU$K_EXIT_OPERATOR_ABORT Literal     76
MENU$K_EXIT_OPERATOR_STOP Literal     76
MENU$K_EXIT_SUP_FILE_LOAD_ERR Literal     76
MENU$K_EXIT_SUP_FILE_NOT_FOUND Literal     76
MM$AUTOSIZER_ERROR  ExtRout  Lib03   507c
MM$BUILD_DDBS   ExtRout  Lib03   500c
MM$BUILD_HSTS   ExtRout  Lib03   501c
MM$CHANGE_TABLE   ExtRout  Lib03   504c
MM$DONE_INITS   ExtRout  Lib03   502c
MM$INTERNAL_ERROR  ExtRout  Lib03   506c
MM$K_ACTION_ADVANCE_STATE Literal     76   450     510     650
MM$K_ACTION_AUTOSIZER_ERROR Literal     76   507     665
MM$K_ACTION_BUILD_DDBS  Literal     76   500     656
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
MM$K_ACTION_BUILD_HSTS  Literal     76   501     657
MM$K_ACTION_CHANGE_TABLE Literal     76   504     662
MM$K_ACTION_DONE_INITS  Literal     76   502     659
MM$K_ACTION_DO_NOTHING  Literal     76   475     509     646
MM$K_ACTION_DUMMY_3  Literal     76   511     647
MM$K_ACTION_DUMMY_4  Literal     76   512     648
MM$K_ACTION_DUMMY_5  Literal     76   513     649
MM$K_ACTION_DUMMY_6  Literal     76   514     655
MM$K_ACTION_DUMMY_7  Literal     76   515     658
MM$K_ACTION_DUMMY_8  Literal     76   516     660
MM$K_ACTION_INTERNAL_ERROR Literal     76   506     664
MM$K_ACTION_LAST_ACTION  Literal     76   494     644
MM$K_ACTION_LOAD_AUTOSIZER Literal     76   497     652
MM$K_ACTION_MENU_PROMPT  Literal     76   505     663
MM$K_ACTION_PRINT_MENU  Literal     76   503     661
MM$K_ACTION_PRINT_MENU_HEADING Literal     76   496     651
MM$K_ACTION_SET_FLAGS_AUTOSIZER Literal     76   498     653
MM$K_ACTION_START_AUTOSIZER Literal     76   499     654
MM$K_STATE_AUTOSIZER_ERROR Literal     76
MM$K_STATE_AUTOSIZER_RUNNING Literal     76
MM$K_STATE_BUILD_DDBS  Literal     76
MM$K_STATE_BUILD_HSTS  Literal     76
MM$K_STATE_CHANGE_TABLE  Literal     76
MM$K_STATE_DONE_INITS  Literal     76
MM$K_STATE_DUMMY_1  Literal     76
MM$K_STATE_DUMMY_2  Literal     76
MM$K_STATE_DUMMY_3  Literal     76
MM$K_STATE_DUMMY_5  Literal     76
MM$K_STATE_DUMMY_7  Literal     76
MM$K_STATE_DUMMY_8  Literal     76
MM$K_STATE_INTERNAL_ERROR Literal     76
MM$K_STATE_LAST_STATE  Literal     76
MM$K_STATE_LOAD_AUTOSIZER Literal     76
MM$K_STATE_MENU_PROMPT  Literal     76
MM$K_STATE_PRINT_MENU  Literal     76
MM$K_STATE_PRINT_MENU_HEADING Literal     76
MM$K_STATE_SET_FLAGS_AUTOSIZER Literal     76
MM$K_STATE_START  Literal     76   120
MM$K_STATE_START_AUTOSIZER Literal     76
MM$LOAD_AUTOSIZER  ExtRout  Lib03   497c
MM$MENU_PROMPT   ExtRout  Lib03   505c
MM$PRINT_MENU   ExtRout  Lib03   503c
MM$PRINT_MENU_HEADING  ExtRout  Lib03   496c
MM$SET_FLAGS_AUTOSIZER  ExtRout  Lib03   498c
MM$START_AUTOSIZER  ExtRout  Lib03   499c
MM_ACTION_ROUTINE_CASE  Macro      345   496     497     498     499     500     501     502     503     504     505
             506     507
    Macro      627   646     647     648     649     650     651     652     653     654     655
             656     657     658     659     660     661     662     663     664     665
MODNAM    Macro    Lib01    92
NUL2ND_    Macro    Lib04    76     235     277     423     431     442     470     646     647     648
             649     650     651     652     653     654     655     656     657     658
             659     660     661     662     663     664     665     669     677     678
```

```
Symbol                          Type  Defined  Referenced ...
------------------------------ -------- ----- ---------------
                679     680     681     682     683     684     685     686     690     698
                699     700     701     702     703     704     705     706     707     708
                709     710     711     712     713     714     715     716     717     718
                719     723     729
RECURSION    Unbound            356     362     381     387     406     412
    Own          329   329=     423.    425=    425.    484=    484.    496=    496.    497=    497.
                498=    498.    499=    499.    500=    500.    501=    501.    502=    502.
                503=    503.    504=    504.    505=    505.    506=    506.    507=    507.
                533=    533.    534=    534.    535=    535.    536=    536.    537=    537.
                560=    560.    562=    562.    563=    563.    564=    564.    565=    565.
                567=    567.    568=    568.    569=    569.    570=    570.    571=    571.
                573=    573.    574=    574.    575=    575.    576=    576.    577=    577.
                578=    578.
STATUS       Unbound            349     351     353     355     358     364     374     376     378     380
                383     389     399     401     403     405     408     414
    Local       496   496=    496.
    Local       497   497=    497.
    Local       498   498=    498.
    Local       499   499=    499.
    Local       500   500=    500.
    Local       501   501=    501.
    Local       502   502=    502.
    Local       503   503=    503.
    Local       504   504=    504.
    Local       505   505=    505.
    Local       506   506=    506.
    Local       507   507=    507.
    Local       533   533=    533.
    Local       534   534=    534.
    Local       535   535=    535.
    Local       536   536=    536.
    Local       537   537=    537.
    Local       560   560=    560.
    Local       562   562=    562.
    Local       563   563=    563.
    Local       564   564=    564.
    Local       565   565=    565.
    Local       567   567=    567.
    Local       568   568=    568.
    Local       569   569=    569.
    Local       570   570=    570.
    Local       571   571=    571.
    Local       573   573=    573.
    Local       574   574=    574.
    Local       575   575=    575.
    Local       576   576=    576.
    Local       577   577=    577.
    Local       578   578=    578.
TQ$CHANGE_TABLE    ExtRout  Lib03   535c
TQ$DONE_TEST_QUEUE  ExtRout  Lib03   536c
TQ$GET_DDB    ExtRout  Lib03   534c
TQ$INIT_TQ    ExtRout  Lib03   533c
TQ$INTERNAL_ERROR  ExtRout  Lib03   537c
```

```
Symbol                              Type  Defined  Referenced ...
----------------------------------- ----- -------- ----------
TQ$K_ACTION_ADVANCE_STATE Literal     76   452      540      678
TQ$K_ACTION_CHANGE_TABLE Literal      76   535      682
TQ$K_ACTION_DONE_TEST_QUEUE Literal   76   536      684
TQ$K_ACTION_DO_NOTHING  Literal       76   477      539      677
TQ$K_ACTION_DUMMY_3  Literal          76   541      679
TQ$K_ACTION_DUMMY_4  Literal          76   542      683
TQ$K_ACTION_DUMMY_5  Literal          76   543      685
TQ$K_ACTION_GET_DDB  Literal          76   534      681
TQ$K_ACTION_INIT_TQ  Literal          76   533      680
TQ$K_ACTION_INTERNAL_ERROR Literal     76   537      686
TQ$K_ACTION_LAST_ACTION  Literal      76   531      675
TQ$K_STATE_CHANGE_TABLE  Literal      76
TQ$K_STATE_DONE_TEST_QUEUE Literal     76
TQ$K_STATE_DUMMY_1  Literal           76
TQ$K_STATE_DUMMY_2  Literal           76
TQ$K_STATE_DUMMY_3  Literal           76
TQ$K_STATE_DUMMY_4  Literal           76
TQ$K_STATE_DUMMY_5  Literal           76
TQ$K_STATE_GET_DDB  Literal           76
TQ$K_STATE_INIT_TQ  Literal           76   150
TQ$K_STATE_INTERNAL_ERROR Literal      76
TQ$K_STATE_LAST_STATE  Literal        76
TQ_ACTION_ROUTINE_CASE  Macro         370   533      534      535      536      537
    Macro       633   677      678      679      680      681      682      683      684      685      686
TYPECODE   Unbound            355      380      405
    RoutForm    282   431.     432.     442.     445.     496.     497.     498.     499.     500.     501.
                502.  503.     504.     505.     506.     507.     533.     534.     535.     536.
                537.  560.     562.     563.     564.     565.     567.     568.     569.     570.
                571.  573.     574.     575.     576.     577.     578.


  CROSS REFERENCE MAP

Line #      Event  File ...
-----  ---------------  ---------
    1  Source (start) DISK$DIAGPACK03:[DS.CRD.V020]MENTURING.B32;1
    2  Module MENTURING
   48  Library #1 DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
   51  Library #2 DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
   54  Library #3 DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
   57  Library #4 SYS$COMMON:[SYSLIB]LIB.L32;2
  733  Eludom MENTURING
```

```
 KEY TO REFERENCE TYPE FLAGS

      . Fetch
      = Store
      c Routine call
      a Address use
      a Indirect use
      e External, external routine, or external literal declaration
      f Forward or forward routine declaration
      h Condition handler enabling
      m Map declaration
      u Undeclare declaration


;     Library Statistics
;
;           -------- Symbols --------     Pages Processing
; File          Total    Loaded   Percent      Mapped Time
;
;  DISK$DIAGPACK03:[DS.LIBRARIES]DS.L32;17
;             671         1         0        46          00:00.1
;  DISK$DIAGPACK03:[DS.LIBRARIES]DIAG.L32;30
;             923         4         0        94          00:00.1
;  DISK$DIAGPACK03:[DS.CRD.V020]CRD.L32;1
;             486        63        12        62          00:00.1
;  SYS$COMMON:[SYSLIB]LIB.L32;2        18619         3          0      1000          00:04.3




;      COMMAND QUALIFIERS

; BLISS MENTURING.B32/LIST=MENTURING.LIS/CROSS_REFERENCE/DEBUG/OBJECT=MENTURING.OBJ/TRACE/OPTIMIZE=(SPEED,LEVEL=3)

; Size:  2116 code + 2076 data bytes
; Run Time:     02:27.4
; Elapsed Time:     02:38.9
; Lines/CPU Min:      298
; Lexemes/CPU-Min: 75708
; Memory Used: 1226 pages
; Compilation Complete
```